Algoritmo de Identificación de Fuente en Imágenes Digitales de Dispositivos Móviles

MÁSTER EN INVESTIGACIÓN EN INFORMÁTICA

Jocelin Rosales Corripio

Director

Luis Javier García Villalba

Calificación: Sobresaliente (10)

Departamento de Ingeniería del Software e Inteligencia Artificial

Facultad de Informática

Universidad Complutense de Madrid

Madrid, Julio de 2013

Algoritmo de Identificación de Fuente en Imágenes Digitales de Dispositivos Móviles

MÁSTER EN INVESTIGACIÓN EN INFORMÁTICA

Jocelin Rosales Corripio

Director

Luis Javier García Villalba

Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid

Madrid, Julio de 2013

La abajo firmante, matriculada en el Máster en Investigación en Informática de la Facultad de Informática, autoriza a la Universidad Complutense de Madrid(UCM) a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a su autor el presente Trabajo Fin de Máster: "Algoritmo de Identificación de Fuente en Imágenes Digitales de Dispositivos Móviles", realizado durante el curso académico 2012-2013 bajo la dirección de Luis Javier García Villalba en el Departamento de Ingeniería del Software e Inteligencia Artificial, y a la Biblioteca de la UCM a depositarlo en el Archivo Institucional E-Prints Complutense con el objeto de incrementar la difusión, uso e impacto del trabajo en Internet y garantizar su preservación y acceso a largo plazo.

Jocelin Rosales Corripio

Luis Javier García Villalba

Agradecimientos

Podría escribir un libro entero con los nombres de las personas con las que estoy agradecida por su impacto positivo en mi vida y por la ayuda que me han brindado para alcanzar mis sueños, a todas ellas gracias.

A mis padres por regalarme la vida y encargarse de que cada minuto la viva plena y feliz. A mis hermanos por ser los mejores compañeros de vida.

A Javier García y a Ana Lucila por la guía, apoyo, confianza, y amistad que me brindaron desde el primer día que los conocí. En especial a Ana por su invaluable ayuda para lograr este trabajo y por recordarme el poder de las palabras.

A los miembros del Grupo de Análisis, Seguridad y Sistemas (GASS) por la amistad, las experiencias que vivimos juntos y el conocimiento compartido.

A mis amigos por ser y estar en cada momento: Daniela, Víctor, Alfredo, Omar, Luis, Martha, José, Angel, Pamela, Mónica y Leonardo.

La presente investigación ha sido financiada por la Fundación General UCM mediante el programa AYUDAS DE POSTGRADO SANTANDER - CONVOCATORIA ECL 2012.

Resumen

Actualmente las imágenes digitales desempeñan un papel importante en nuestra sociedad. La presencia de dispositivos móviles con cámaras fotográficas integradas crece a un ritmo imparable, provocando que la mayoría de las imágenes digitales provengan de este tipo de dispositivos. El desarrollo tecnológico no sólo facilita la generación de estas imágenes, sino también la manipulación intencionada de éstas. Las técnicas de análisis forense de imágenes de dispositivos móviles cobran, pues, especial importancia. En este trabajo se propone una serie de algoritmos basados en el ruido del sensor y en la transformada wavelet que permiten identificar el dispositivo móvil (marca y modelo) que generó la imagen, eliminar la posibilidad de identificación del mismo y falsificar la identidad de una imagen dada.

Palabras clave: Análisis Forense, Clasificación, Identificación de Fuente, Imágenes Digitales, Máquina de Soporte Vectorial, PRNU, Respuesta Fotónica No Uniforme, Ruido del Sensor, SVM, Transformada Wavelet.

Abstract

Nowadays digital images play an important role in our society. The mobile device camera presence is growing at an unstoppable rate, causing that most of digital images come from this kind of devices. While the developing technology makes image generation process easier, at the same time it facilitates forgery; therefore, image forensics is gaining relevance. This work proposes a group of algorithms based on sensor noise and the wavelet transform that allows identifying the mobile device (brand and model), which has generated a picture; eliminate the possibility of identifying the source mobile device camera; and finally forge the identity of an image.

Keywords: Classification, Digital Image, Forensics Analysis, Photo Response Non Uniformity, PRNU, Sensor Imperfection, Source Model Identification, Support Vector Machines, SVM, Wavelet Transform.

Índice General

Índice de Figuras

Índice de Tablas

Lista de Acrónimos

ADC	*Analog Digital Conversion*.
API	*Application Programming Interface*.
CCD	*Charge Coupled Device*.
CFA	*Color Filter Array*.
CMOS	*Complementary Metal Oxide Semiconductor*.
CMY	*Cyan-Magenta-Yellow*.
CYYM	*Cyan-Yellow-Yellow-Magenta*.
DBC	*Decision Boundary Carving*.
DCT	*Discrete Cosine Transform*.
DIP	*Digital Image Processor*.
DSC	*Digital Still Camera*.
EM	*Expectation-Maximization*.
EXIF	*Exchangeable Image File Format*.
FAR	*False Acceptance Rate*.
FPN	*Fixed Pattern Noise*.
GPS	*Global Positioning System*.

GRGB *Green-Red-Green-Blue.*

IFD *Image File Directory.*

IID *Independent and Identically Distributed random variable.*

IQM *Image Quality Metrics.*

JPEG *Joint Photographic Experts Group.*

MAP *Maximum A-Posteriori Probability.*

MOS *Metal Oxide Semiconductor.*

PCE *Peak to Correlation Energy.*

PNU *Pixel Non-Uniformity.*

PRNU *Photo Response Non Uniformity.*

PSD *Photoshop Data file.*

QMF *Separable Quadrate Mirror Filters.*

RAT *Radon Transform.*

RBF *Radial Basis Function.*

RGB *Red-Green-Blue.*

RGBE *Red-Green-Blue-Emerland.*

ROI *Region Of Interest.*

SFFS *Sequential Forward Featured Selection.*

SFS *Sequential Floating Search.*

SPN *Sensor Pattern Noise.*

SVM *Support Vector Machine*.

TIFF *Tagged Image File Format*.

XMP *eXtensible Metadata Platform*.

Capítulo 1

Introducción

Con frecuencia las fotografías son consideradas como una parte de la verdad al ser hechos reales capturados por dispositivos electrónicos (cámaras). Sin embargo, con el desarrollo de la tecnología han surgido herramientas potentes y sofisticadas que facilitan de una manera impresionante la alteración de las imágenes digitales, incluso para quienes no tienen conocimientos técnicos o especializados en el área [GKWB07].

El desarrollo de las tecnologías digitales ha estado y continúa avanzando a un ritmo imparable. Cada día el número de cámaras digitales va creciendo, así como la facilidad de acceso a ellas. Las cámaras digitales de móviles merecen especial atención, ya que estudios realizados indican que al final del año 2012 el número total de dispositivos móviles activos alcanzó los 6,7 billones y en este verano (2013) se vivirá lo que algunos llaman "El momento del móvil" ya que por primera vez una tecnología de consumo igualará al total de la población del planeta (7,1 billones).

El 83 % de estos móviles cuentan con cámara digital integrada, las cuales a diferencia de las cámaras digitales convencionales son llevadas por sus dueños todo el tiempo a la mayoría de lugares que asiste y, en muchos casos, estos dispositivos tienen conexión a internet.

Debido al incremento en sus capacidades de almacenamiento, de procesamiento, de usabilidad y de portabilidad así como a su bajo coste, los dispositivos móviles están presentes en diversidad de actividades, lugares y eventos de la vida diaria. Algunos datos que permiten hacerse una idea de la magnitud de la presencia de este tipo de dispositivos son los siguientes [AM12]:

- Más del 90 % de las personas que alguna vez ha tomado una fotografía lo ha hecho únicamente con cámaras de dispositivos móviles.
- Un gran número de personas tienen y usan más de un dispositivo móvil.
- Las estadísticas globales arrojan que un usuario típico en promedio mira su móvil 150 veces al día y 8 de ellas es para hacer uso de la funcionalidad de la cámara.

A causa del extenso uso de las cámaras digitales de dispositivos móviles se han generado

polémicas, discusiones y normas sobre la prohibición de su uso en lugares como escuelas, oficinas de gobierno, eventos empresariales, conciertos, empresas, etc.

Una consecuencia más de su extenso uso es que las imágenes digitales en la actualidad son utilizadas como testigos silenciosos en procesos judiciales, siendo una pieza crucial de la evidencia del crimen [AZ06]. Es por ello que contar con herramientas que permitan identificar a los dispositivos que han generado una cierta imagen digital cobra importancia ya que podría servir en diversas áreas como la lucha contra la pornografía infantil, la prevención de robo de tarjetas de crédito, el combate a la piratería, la prevención de secuestros, etc.

De manera análoga a la balística que trata de relacionar una pistola con sus balas, el análisis forense de imágenes digitales trata de identificar la imagen con la cámara digital con la que fue generada [WGKM09].

1.1 Objeto de la Investigación

La fuente de una imagen digital se puede identificar a través de los rasgos que el dispositivo que la genera impregna en ella durante el proceso de su generación.

La mayoría de las investigaciones realizadas en los últimos años sobre técnicas de identificación de fuente se han enfocado únicamente en la identificación de cámaras tradicionales *Digital Still Camera* (DSC). Considerando que hoy en día las cámaras de los dispositivos móviles prácticamente han sustituido a las DSCs se detectó la necesidad de realizar investigación sobre las técnicas para identificar la fuente de imágenes generadas por dispositivos móviles.

Las imágenes digitales generadas por un dispositivo (móvil o no) contienen intrínsecamente un patrón del ruido del sensor que se puede usar como medio de identificación de la fuente [LFG06]. Específicamente, las cámaras digitales de dispositivos móviles cuentan en su mayoría con un tipo de sensor que deja rasgos característicos en la imagen. La presente investigación se centra en las técnicas de identificación de fuente basadas en el ruido del sensor.

Este trabajo propone un algoritmo que mejora la identificación de los dispositivos móviles fuente de una imagen. Asimismo, propone un algoritmo para falsificar la huella de una imagen que no requiere acceso a la cámara del dispositivo fuente.

1.2 Trabajos Relacionados

Las tareas de análisis forense de imágenes digitales se dividen, de acuerdo a su objetivo, en las siguientes ramas: verificación de integridad, recuperación de la historia de procesamiento, clasificación basada en la fuente, agrupación por dispositivo fuente e identificación de la fuente [CFGL08].

Para el diseño de técnicas y algoritmos en cualquiera de estas ramas se aprovechan algunas características especiales de las imágenes creadas con móviles que sirven como

herramienta para el análisis forense. En [VCEK07, TNC10] se realiza un estudio de las características que pueden ser objeto de análisis forense en dispositivos móviles.

Existe una gran variedad de trabajos que hacen referencia a los distintos tipos de metadatos en las imágenes con fines de búsqueda de información, clasificación de imágenes e identificación de la fuente [BL04, BL05, Tes05, RCC+08, Are11]. Estos trabajos buscan patrones en los metadatos de las fotografías que arrojen datos de interés para el análisis forense.

La gran desventaja del uso de metadatos es su facilidad de manipulación. Existen aplicaciones que además de permitir la consulta de los metadatos también posibilitan su edición y eliminación de una forma sencilla.

Debido a la vulnerabilidad de los metadatos es necesario ir más allá diseñando técnicas y algoritmos que utilicen el contenido de la imagen. Al igual que ocurre con los metadatos, las imágenes pueden ser modificadas malintencionadamente para evitar las técnicas forenses que se les apliquen. Sin embargo, las técnicas y algoritmos basados en el contenido de la imagen son más robustos ya que se requiere de un mayor nivel de conocimientos para impedir el análisis que se les realiza.

Para el diseño y creación de estos algoritmos es necesario tener un amplio conocimiento del proceso de generación de imágenes por parte de este tipo de dispositivos. En [MSGW08] se describe el proceso de adquisición de imágenes en cámaras de dispositivos móviles haciendo asimismo una comparativa de este proceso frente al existente en DSCs y en escáneres.

Este trabajo se centra en la rama de la identificación de la fuente con la que se genera una imagen, y pretende obtener la marca y modelo del dispositivo que genera una imagen dada. Los estudios realizados hasta el momento en este área se dividen básicamente en cuatro grupos dependiendo de la información que se utiliza como base para identificar la fuente [VCEK07].

El primer grupo utiliza la información de la aberración de las lentes en la cámara [Cho06, CLW06].

El segundo grupo tiene en cuenta el proceso de interpolación propio de cada dispositivo que se da a partir de su matriz *Color Filter Array* (CFA) [BSM06, CAS+06, LH06, BSM08, CK09].

El tercer grupo se basa en características de las imágenes válidas para el estudio forense, separando éstas en tres grandes grupos: características del color (*color features*), características de la calidad (*quality features*) y características de la imagen en el dominio de la frecuencia (*image characteristics of frequency domain*). Existen referencias concretas sobre el tratamiento de cada conjunto de características [AMS03, AKMS05, LF06, TLL07, MSGW08, MKY08, WGKM09, OA11, LLC+12].

El cuarto grupo, y objeto de investigación en este trabajo, utiliza las características basadas en el ruido del sensor y en la información de la matriz del sensor *Charge Coupled Device* (CCD) o *Complementary Metal Oxide Semiconductor* (CMOS) [GBK+01, LFG06, CESR12].

Una vez seleccionada el tipo de información que se va a utilizar para el análisis, se ha de elaborar un algoritmo que dada una imagen nos ofrezca la identificación de la fuente con el mayor grado de fiabilidad posible.

Existen distintos algoritmos utilizados para DSCs que pueden adaptarse a nuestra investigación [GBK+01, MSM04, BL05, CLW06, LH06, Cho06, LFG06, BSM08, LH06, BSM06, LFG06, DSM07, MKY08, WGKM09, OA11].

Asimismo, existen algoritmos y técnicas específicos para dispositivos móviles [TLL07, TLL07, CSA08, MSGW08, LC09, LLC+12]. Sin embargo, estas técnicas no hacen uso de la información proveniente de las imperfecciones del sensor que, debido a las características de los dispositivos móviles, resultan ser las más adecuadas.

1.3 Estructura del Trabajo

El resto del trabajo está organizado en 6 capítulos y 4 anexos con la estructura que se comenta a continuación: El Capítulo 2 introduce algunos conceptos que son elementales para comprender el análisis forense en imágenes. Así, se describe el proceso de formación de una imagen digital, los elementos de la cámara que sirven de base para las técnicas forenses en imágenes, los tipos de sensores y el ruido que generan en las imágenes. Por último, se remarcan las diferencias entre las cámaras tradicionales y las cámaras de dispositivos móviles.

El Capítulo 3 comienza mostrando las diferentes tareas que engloba el análisis forense de imágenes. A continuación, se presenta un estado del arte de las técnicas forenses en imágenes para la identificación de la fuente, agrupándolas de acuerdo a las características que usan para realizar esta tarea. De cada grupo se incluye en orden cronológico una descripción de las diferentes propuestas que han surgido. En este capítulo se incluye una tabla comparativa que resume las principales características de cada propuesta analizada. Asimismo, se presentan los ataques a las técnicas forenses de imágenes, clasificándolas de acuerdo a sus objetivos. Así, se describe en primer lugar una técnica utilizada para ocultar cuando una imagen se ha procesado después de su formación. Posteriormente, se muestran dos ataques a las técnicas de identificación de fuente describiendo un algoritmo para destruir la huella del sensor y otro para suplantar una huella en una imagen víctima. Se finaliza con la descripción de un método para detectar cuando se ha llevado a cabo la falsificación de identidad de una imagen.

El Capítulo 4 presenta las contribuciones de este trabajo. Así, en primer lugar, se presenta un algoritmo basado en los rasgos del sensor y en la transformada *wavelet* para la identificación de la marca y modelo del dispositivo móvil fuente de una imagen. Posteriormente, se especifica un algoritmo para llevar a cabo la falsificación de la identidad de una imagen.

El Capítulo 5 describe los experimentos realizados para evaluar la efectividad de los algoritmos propuestos en el capítulo 4 y presenta los resultados obtenidos.

El Capítulo 6 muestra las principales conclusiones de este trabajo, las líneas futuras

de investigación y las publicaciones derivadas del presente trabajo.

El Anexo A contiene una breve introducción a la Transformada *Wavelet* y a sus aplicaciones en el análisis forense de imágenes.

En el Anexo B se describe la librería *Support Vector Machine* (SVM) utilizada para la implementación de la identificación de la fuente.

El Anexo C describe los lenguajes y librerías utilizados para la implementación de los algoritmos.

Por último, el Anexo D estudia los sistemas de clasificación. En la primera parte se describen los elementos que componen un sistema de clasificación y los diferentes tipos de aprendizaje. Después, se muestra la taxonomía de los clasificadores. Finalmente, se describen detalladamente las máquinas de soporte vectorial SVM pues han sido de gran importancia en el proceso de identificación de la fuente de imágenes digitales.

Capítulo 2

Análisis Forense de Imágenes Digitales

El objetivo de este capítulo es mostrar cómo se genera una imagen digital, así como describir los componentes que participan en este proceso. Estos conceptos son la base de las técnicas de análisis forense descritas en los siguientes capítulos.

2.1 Formación de una Imagen Digital

labelsection21 Para comprender el análisis forense de las imágenes digitales lo primero que se requiere conocer es cómo está compuesta una cámara fotográfica y cuál es el procedimiento que realiza para generar una imagen (a menudo llamado *pipeline*). Las cámaras fotográficas se componen de un sistema de lentes, un grupo de filtros, una matriz de filtro de colores o CFA, un sensor de imagen y un procesador de imagen o *Digital Image Processor* (DIP) [BSM08]. A pesar de que muchos de los detalles del *pipeline* se mantienen como información confidencial de los fabricantes este proceso es muy similar en la mayoría de las cámaras digitales. La estructura básica se muestra en la Figura 2.1.

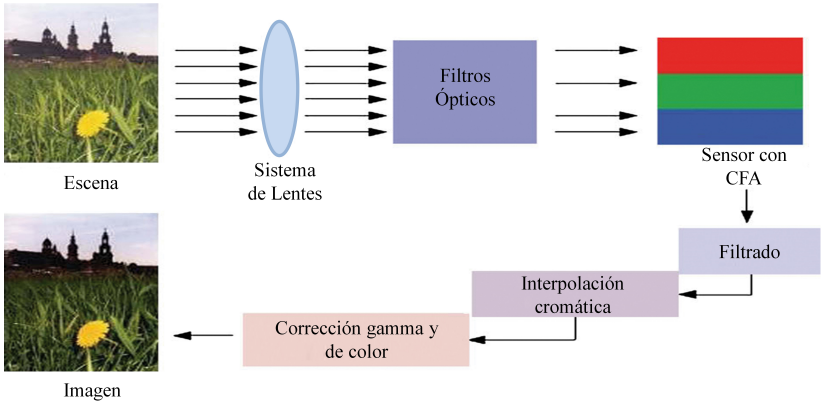


Figura 2.1: Proceso de adquisición de imágenes en cámaras digitales

Como primer paso para generar una imagen el sistema de lentes captura la luz de la escena controlando la exposición, el foco y la estabilización de la imagen. Después, la luz que entra en la cámara a través del sistema de lentes pasa por un grupo de filtros que mejora la calidad visual de la imagen. Este grupo incluye al menos un filtro infrarrojo y un filtro *anti-aliasing*. El filtro infrarrojo absorbe o refleja la luz permitiendo que sólo la parte visible del espectro pase a la siguiente fase, evitando que la radiación infrarroja ocasione pérdida de nitidez en la imagen. El filtro *anti-aliasing* se encarga de limpiar la señal produciendo imágenes con contornos más suaves.

A continuación la luz pasa al sensor de la imagen que es una matriz de elementos sensibles a la luz llamados píxeles. Cada elemento de esta matriz de píxeles integra la luz incidente y genera una señal analógica proporcional a la intensidad de la luz recibida. Esta señal se convierte en una señal digital y se transmite al procesador de imagen. Debido a que el sensor de la imagen es monocromático, para capturar una imagen a color se requieren diferentes sensores. Idealmente, un sensor para cada color. Sin embargo, debido al coste que esto implica, en la mayoría de las cámaras sólo se usa un sensor de imagen junto a una matriz de filtros de color que se coloca antes del sensor para producir los colores.

Una vez que el procesador de imagen recibe la señal digital generada por el sensor elimina el ruido y otras anomalías introducidas en las señales digitales (*artifacts*), con la finalidad de obtener una imagen visualmente agradable. Uno de los procesos que se realizan sobre la señal es la llamada interpolación cromática (*demosaicing*) encargada de calcular los valores de los colores faltantes debido a que el sensor únicamente proporciona información sobre una cierta cantidad de colores (los que permite pasar la matriz de filtros de color). Un proceso adicional es la corrección de píxeles defectuosos originados por imperfecciones en el sensor, que corrige estos píxeles mediante interpolación.

Otro proceso al que se somete la imagen es el balanceo de blancos, que permite una reproducción más fiel del color, evitando que haya colores dominantes. Por último, el proceso de corrección *gamma* ajusta los valores de intensidad de la imagen. Aunque los algoritmos para llevar a cabo estos procesos están presentes en todas las cámaras, los detalles exactos de la forma de realizarlos pueden variar entre los diferentes fabricantes e, incluso, entre los modelos de un mismo fabricante.

Finalmente, la imagen generada por el procesador de imagen se comprime. En las cámaras de dispositivos móviles normalmente se utiliza el algoritmo *Joint Photographic Experts Group* (JPEG) [Ham] para ahorrar espacio, almacenándose en la memoria del dispositivo junto con la información de la imagen en formato *Exchangeable Image File Format* (EXIF) [RSYD05].

2.1.1 Filtros de Color

La matriz de filtros de color es una de las partes más importantes de la cadena de procesamiento para la generación de una imagen de las cámaras de un solo sensor [APS98]. La CFA se encuentra sobre el sensor monocromo, y su función es adquirir la información del

color de la escena. Cada celda del filtro de color deja pasar la luz de acuerdo a un rango de longitudes de onda, de tal manera que las intensidades filtradas separadas incluyen información sobre el color de la luz. Como se ilustra en la Figura 2.2, la intensidad de la luz que pasa por cada una de las celdas forma una imagen en escala de grises y, dependiendo de la configuración del filtro CFA, se interpreta como una imagen a color (considerando que cada píxel corresponde a un valor de intensidad).

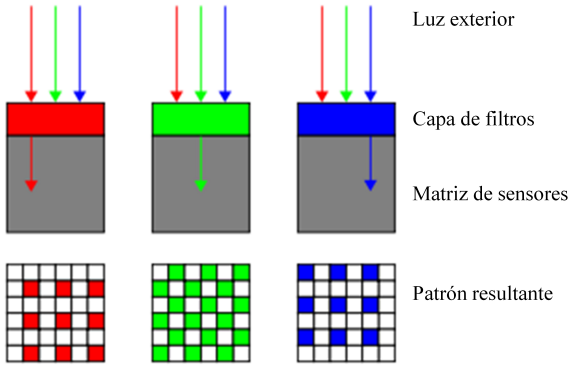


Figura 2.2: Matriz de filtros de color (CFA)

En este punto el proceso la interpolación cromática se lleva a cabo para obtener los valores que faltan para cada uno de los colores del filtro CFA. Este proceso es el más complejo en cuanto a cómputo se refiere. Su algoritmo utiliza los valores de los píxeles vecinos para obtener todos los valores que no han sido medidos.

Es posible que los tipos de filtros de color y la forma de realizar la interpolación cromática varíen entre fabricantes. Además, existe la posibilidad de que la imagen se almacene en formato CFA y el proceso de interpolación cromática se realice en un ordenador, extendiendo aún más la posibilidad de variaciones en este proceso.

El diseño de la matriz CFA utilizada influye en la imagen resultante de la cámara, tanto en la nitidez y apariencia de los bordes como en los pequeños detalles. A veces, el proceso de interpolación cromática puede generar anomalías en la imagen tales como el *aliasing* (efecto que produce el aspecto desagradable de líneas escalonadas "sierras" en los contornos de las imágenes), ruido y distorsiones en el color. El uso de otro filtro puede eliminar la presencia de estas imperfecciones en determinadas áreas de la imagen a costa de degradar la calidad en otras [LP05].

Generalmente, las cámaras usan el modelo *Green-Red-Green-Blue* (GRGB) del patrón CFA de Bayer. La salida de un sensor de este tipo es un mosaico de píxeles rojos, verdes y azules de diferentes intensidades. Como se observa en la Figura 2.2, este filtro captura el 25 % de los píxeles en el canal rojo, el 50 % en el canal verde y el 25 % restante en el canal azul. Otras alternativas de filtros CFA son los patrones *Cyan-Yellow-Yellow-Magenta* (CYYM), *Red-Green-Blue-Emerland* (RGBE) y *Cyan-Magenta-Yellow* (CMY).

2.1.2 Tipos de Sensores

El sensor de la imagen es la parte más importante de las cámaras digitales. Generalmente, se considera el corazón de la cámara. Éste es una matriz de elementos sensibles a la luz llamados píxeles. Los píxeles están hechos de silicio y capturan la luz convirtiendo los fotones en electrones utilizando el efecto fotoeléctrico. Cada píxel se encarga de acumular la carga inducida por la luz durante un determinado tiempo de exposición para luego ser leído y procesado. La señal de salida del sensor es proporcional a la carga acumulada, dependiendo de la cantidad de luz que incida sobre el píxel y del tiempo de exposición a ella.

Existe una extensa literatura sobre el desarrollo y tecnologías de los sensores [HL07, Nak05, HKT07, AP13]. Sin embargo, para el propósito de este trabajo basta con tener una visión general para comprender el ruido que el sensor puede introducir en las imágenes que genera.

Los sensores de la imagen se agrupan de acuerdo a sus procesos de fabricación en CCD y CMOS [HL07]. Los dos tipos de sensores están formados esencialmente por semiconductores de metal-óxido *Metal Oxide Semiconductor* (MOS) distribuidos en forma de matriz y funcionan de una manera muy similar. Sin embargo, hay características que diferencian a estas tecnologías.

2.1.2.1 Sensores CCD

La diferencia clave entre las dos tecnologías de sensores es el lugar en el que se digitalizan los píxeles y la forma en la que se lleva a cabo la lectura de las cargas.

En el caso de los sensores CCD cada una de las cargas de las celdas de la matriz se transforman en voltajes y se entrega una señal analógica como salida para que posteriormente se digitalice por la cámara. La estructura de este tipo de sensores es muy sencilla, pero tiene como inconveniente la necesidad de contar con un chip adicional que trate la información de salida del sensor (implicando equipos más grandes y costosos).

A diferencia de los sensores CMOS que soportan la lectura de la matriz de píxeles de una manera aleatoria, en los sensores CCD todos los píxeles comienzan y finalizan la integración de carga al mismo tiempo. Esto propicia una salida uniforme (resultado que se espera de un píxel sometido al mismo nivel de excitación de los demás sin que se presenten cambios notables en la señal obtenida). A este tipo de exposición se le conoce como *global shutter*. Es posible añadir circuitos en los sensores de CMOS para hacer que den un resultado similar. Sin embargo, siguen estando sobre ellos los sensores de tipo CCD.

Los sensores del tipo CCD son, por mucho, mejores que los de tipo CMOS en cuanto al rango dinámico (coeficiente entre la saturación de los píxeles y el umbral por debajo del cual no captan señal), puesto que al ser menos sensibles toleran mejor los extremos de luz. Asimismo, los sensores CCD son superiores a los CMOS en términos de ruido en la imagen, puesto que el procesado de las señales se lleva a cabo en un chip externo que puede optimizarse para el desarrollo de esta función. En contraste, los sensores CMOS

realizan el procesamiento de la señal dentro del mismo sensor dejando menos espacio para colocar los foto-diodos encargados de recolectar la luz.

2.1.2.2 Sensores CMOS

Los sensores CMOS son sensores con un diseño de píxeles activos e independientes. Se denominan píxeles activos debido a que la digitalización se realiza en ellos internamente en unos transistores que ofrecen mejor velocidad de procesamiento, eliminándose la necesidad de un chip externo que realice esta función, lo que reduce el coste y el tamaño de los equipos.

La característica de independencia se refiere a la flexibilidad que este tipo de sensores ofrece para la lectura de la matriz de píxeles, ya que es posible acceder a cada celda mediante la posición de su fila y columna. Generalmente, la lectura de la matriz se realiza en forma de barrido progresivo. A este esquema se le conoce como *roolling shutter* (no es necesario leer la matriz completa en un solo tiempo como en los sensores CCD). Además, al estar formados por celdas independientes, los sensores CMOS no presentan el efecto *blooming*. Este efecto se produce cuando un píxel se satura por la luz que incide sobre él y a continuación comienza a saturar a los que se encuentran a su alrededor.

Una ventaja más es que los sensores CMOS son más sensibles a la luz y en condiciones de poca iluminación se comportan mejor. Adicionalmente, debido a que los amplificadores de la señal se encuentran dentro de la misma celda, no se genera un consumo extra de alimentación a diferencia de los sensores CCD.

En sus inicios los sensores CMOS no eran considerados tan buenos como los sensores CCD. Sin embargo, la tecnología CCD ha llegado a su límite y ahora es cuando se está desarrollando la tecnología CMOS superando sus deficiencias [CCD]. La mayoría de las cámaras utilizan sensores CCD, aunque en dispositivos móviles es más común el uso de sensores CMOS.

2.1.3 Imperfecciones y Ruido de la Imagen

2.1.3.1 Imperfecciones del Sensor

Durante el proceso de generación de una imagen es posible que se introduzcan algunos defectos que se vean reflejados como ruido en la imagen final.

Teniendo una noción básica del funcionamiento de los sensores, se pueden analizar los defectos que generan ruido en las imágenes finales. Estos defectos son de gran ayuda para identificar la cámara que generó una imagen determinada.

Se consideran defectos los píxeles que tienen una respuesta lo suficientemente anormal como para ser descartados y no formar parte de los datos de la imagen final. A pesar de que en la cadena de procesamiento de la imagen se realizan procedimientos para tratar de mitigar estos defectos, las correcciones pueden no ser del todo perfectas y es posible inyectar defectos ocultos en la imagen; incluso algunos defectos son tolerados por los fabricantes con tal de mantener o mejorar el rendimiento de las cámaras.

De acuerdo a los factores que los ocasionan, los defectos se pueden agrupar en:

- **Defectos de fila y columna**: Pueden ser ocasionados durante el proceso de transferencia de carga. Debido a la forma en que los sensores CMOS direccionan los píxeles se pueden generar errores parciales o totales en filas o columnas de píxeles.

- **Defectos de grupo** : Este tipo de defectos afectan a un conjunto de píxeles. Pueden ser ocasionados por defectos en la superficie del sensor como suciedad o rayas. También pueden ser causados por fallos eléctricos como es el caso de algunos sensores de tipo CMOS en los que múltiples píxeles (generalmente 3 ó 4) comparten circuitería para convertir la carga en voltaje, y al haber un fallo en alguno de estos píxeles se produce un defecto en grupo.

- **Píxeles calientes**: Son los píxeles que generan altas salidas de voltaje bajo cierto tipo de condiciones, especialmente en exposiciones largas. Los puntos que se obtienen en la imagen final son siempre muy brillantes y puede ser de cualquier color, dependiendo del punto del patrón Bayer que esté precisamente frente al píxel en cuestión.

- **Píxeles muertos**: Son los píxeles que tienen una respuesta muy pobre a la luz, apareciendo como puntos negros en las imágenes finales.

- **Diferencias entre salidas múltiples**: En los sensores que tienen más de una salida pueden presentarse variaciones entre las diferentes salidas, especialmente si utilizan un convertidor analógico/digital para cada una de las salidas. Los rasgos creados en la imagen por la falta de coincidencia tendrán una textura dependiendo de la disposición geométrica de los píxeles canalizados a través de cada salida.

- **Interferencia**: Este defecto se produce cuando los fotones que deberían de ser recolectados por un píxel se recogen por un píxel vecino. La mayoría de sensores sufren este tipo de defecto que puede ser causado por problemas de reflexión en el sistema de lentes o por la difusión de la carga durante exposiciones a la luz prolongadas.

- **Saturación**: Sucede cuando un píxel acumula más carga de la que puede contener y el exceso de la carga es pasada a los píxeles vecinos generando el efecto *blooming*.

- **Rolling Shutter**: La técnica de *rolling shutter* utilizada en los sensores CMOS puede crear distorsiones en la imagen cuando la escena cambia significativamente mientras está siendo capturada como cuando hay movimientos en la escena (deformando la imagen) o cambios de iluminación (introduciendo líneas en la misma).

- **Corriente de oscuridad**: Surge de las impurezas del cristal de silicio de los sensores. Es la señal acumulada en cada píxel incluso en la ausencia de luz y que varía además con la temperatura [Nak05]. En los sensores pequeños estas variaciones son muy pequeñas, pero en los sensores de mayor tamaño suelen ser más significativas.

2.1.3.2 Ruido en la Imagen

Existen diversas fuentes de imperfecciones y ruido introducidas en las diferentes etapas del proceso de generación de la imagen en la cámara. Incluso si se toma una fotografía uniforme y completamente iluminada es posible observar pequeños cambios de intensidad entre los píxeles. Esto se debe al ruido de disparo que es aleatorio y, en gran parte, al patrón de ruido que es determinista y se mantiene aproximadamente igual si se toman varias fotografías de la misma escena.

El patrón de ruido en una imagen se refiere a cualquier patrón espacial que no cambia de una imagen a otra y está compuesto por el ruido espacial que es independiente de la señal o ruido de patrón fijo *Fixed Pattern Noise* (FPN) y el ruido espacial debido a la diferencia de respuesta de cada píxel a la señal incidente o ruido de respuesta no uniforme *Photo Response Non Uniformity* (PRNU) [KMC+06, LFG06, AP13]. La estructura del patrón de ruido se ilustra en la Figura 2.3.

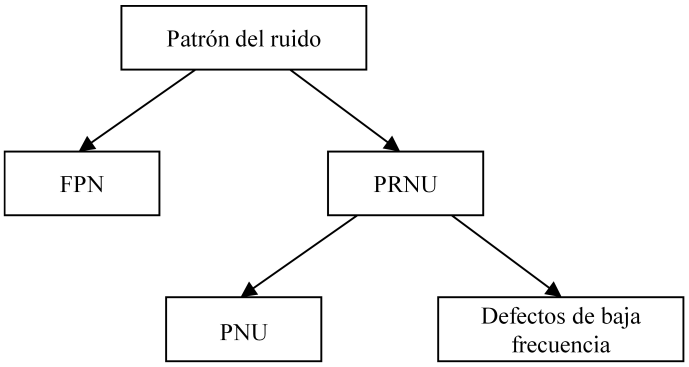


Figura 2.3: Patrón de ruido del sensor

El ruido FPN se genera por la corriente de oscuridad y también depende de la exposición y de la temperatura. Debido a que el ruido del patrón fijo es un ruido independiente aditivo, algunas cámaras lo eliminan automáticamente restando un marco oscuro a las imágenes que generan.

El ruido PRNU es la parte dominante del patrón de ruido de las imágenes y es un ruido dependiente multiplicativo. El ruido PRNU está formado principalmente por la uniformidad de pixel *Pixel Non-Uniformity* (PNU) y los defectos de baja frecuencia como la configuración del *zoom* y la refracción de la luz en las partículas de polvo y lentes.

El ruido PNU es la diferencia de sensibilidad a la luz entre los píxeles de la matriz del sensor. Se genera por la falta de homogeneidad de las obleas de silicio y las imperfecciones durante el proceso de fabricación del sensor. Debido a su naturaleza y origen es muy poco probable que incluso los sensores procedentes de la misma oblea presenten patrones PNU correlacionados. Este ruido no se ve afectado por la temperatura ambiente ni por la humedad.

El ruido PNU es normalmente más común, complejo y significativo en los sensores de

tipo CMOS debido a la complejidad de la circuitería de la matriz de píxeles.

2.1.4 Diferencia entre Cámaras Digitales y Cámaras de Dispositivos Móviles

A pesar de que las cadenas de procesamiento de la imagen entre las cámaras digitales tradicionales y las cámaras de dispositivos móviles son muy parecidas, existen algunas diferencias significativas en cuanto a calidad entre las cámaras [CSA08].

Las cámaras de dispositivos móviles generan imágenes de menor calidad debido a varios factores relacionados principalmente con el *hardware* que utilizan dada la naturaleza compacta de este tipo de dispositivos. Estos factores son:

- **Apertura de la lente**: Restringida a tener valores pequeños para la apertura de la lente.
- **Resolución**: Las resoluciones son menores.
- **Distancia focal**: Tienen una distancia focal fija y restringida a valores pequeños que limita las condiciones de iluminación.
- **Flash**: Muchos de los dispositivos móviles no cuentan con *flash* y, en caso de tenerlo, no es muy robusto debido a las limitaciones de potencia. La sensibilidad a la luz del sensor de acuerdo a su tipo afecta directamente a la velocidad de obturación, y esto se relaciona con la falta de definición de la imagen.
- **Conversión Analógica Digital**: Los dispositivos móviles están limitados al uso de *Analog Digital Conversion* (ADC) de 10 bits mientras que las cámaras tradicionales típicamente usan uno de 12 bits.

Como se muestra en [CSA08] las características generadas por las cámaras digitales tradicionales y las de dispositivos móviles son diferentes. En los resultados de los experimentos realizados sólo un 49,5 % de las características resultaron ser comunes y, por lo tanto, no son intercambiables.

Las huellas CFA son más prominentes en las cámaras digitales tradicionales mientras que las cámaras de dispositivos móviles tienen una mayor contaminación de ruido debido a los factores mencionados anteriormente y a la diferencia de calidad entre los sensores CMOS y CCD. Es por ello que las técnicas de identificación de fuente basadas en el ruido del sensor y las que se basan en la transformada *wavelet* resultan ser más adecuadas en dispositivos móviles.

Capítulo 3

Técnicas de Análisis Forense en Imágenes

En este capítulo se describen las principales técnicas de análisis forense de imágenes digitales haciendo énfasis en las técnicas de identificación de la fuente de la imagen, ya que es la rama del análisis forense en la que se centra este trabajo.

Según [CFGL08] las tareas de análisis forense de imágenes digitales se pueden dividir en las siguientes categorías:

- **Verificación de integridad o detección de falsificaciones**: Busca descubrir procedimientos maliciosos que se hayan aplicado a las imágenes como, por ejemplo, recorte o adición de objetos a una imagen.
- **Recuperación de la historia de procesamiento**: Tiene como objetivo recuperar la cadena de procesamientos que han sido aplicados a una imagen de una manera no maliciosa como, por ejemplo, recortes, filtrados, contrastes, etc.
- **Clasificación basada en la fuente**: Tiene como objetivo clasificar las imágenes de acuerdo a su origen en cámaras digitales o escáneres.
- **Agrupación por dispositivos fuente**: Dado un grupo de imágenes se buscan los grupos de imágenes que fueron obtenidas utilizando la misma cámara.
- **Identificación de la fuente**: Busca determinar el dispositivo que generó una imagen determinada.

3.1 Técnicas de Identificación de la Fuente

La investigación en este campo estudia el diseño de técnicas para identificar las características, especialmente marca y modelo, de los dispositivos utilizados para la generación de imágenes digitales.

El éxito de estas técnicas depende del supuesto de que todas las imágenes adquiridas por un mismo dispositivo presentan características intrínsecas del dispositivo. Las características que se usan para identificar marca y modelo de las cámaras digitales se derivan de las diferencias que existen entre las técnicas de procesamiento de las imágenes y las tecnologías de los componentes que se utilizan. El mayor problema con este enfoque es que los diferentes modelos de las cámaras digitales usan componentes de un número reducido de fabricantes, y que los algoritmos que usan también son muy similares entre modelos de la misma marca. Es por ello que la fiabilidad de la identificación de la cámara fuente depende en gran parte de la identificación de varias características independientes del modelo. Según [VCEK07] se pueden establecer cuatro grupos de técnicas para este fin: utilización de la aberración de las lentes, interpolación de la matriz CFA, uso de las características de la imagen e imperfecciones del sensor. Esta última constituye el objeto de este trabajo. Además de las anteriores existe otro grupo de técnicas basadas en los metadatos.

3.1.1 Técnicas Basadas en Metadatos

Las cámaras digitales cuentan con una poderosa fuente de información que son los metadatos embebidos en los archivos de las imágenes digitales que generan. Los metadatos o "datos sobre datos" registran información relacionada con las condiciones de captura de la imagen, como fecha y hora de generación, presencia o ausencia de *flash*, distancia de los objetos, tiempo de exposición, apertura del obturador, *Global Positioning System* (GPS), entre otros. En otras palabras, información de interés que complementa el contenido principal de un documento digital. Los metadatos pueden llegar a ser una potente ayuda para la organización y búsqueda a lo largo de librerías de imágenes.

Las imágenes digitales son almacenadas en una gran variedad de formatos como *Tagged Image File Format* (TIFF) [Ass],JPEG [Ham] y *Photoshop Data file* (PSD) u otros propietarios como RAW. Algunos de los distintos contenedores de metadatos para los distintos formatos son: *Image File Directorys* (IFDs) EXIF/TIFF, Adobe *eXtensible Metadata Platform* (XMP) [XMP] e IPTC-IIM [IPT]. La especificación EXIF [Comb] es la más utilizada para identificación de la fuente por ser el contenedor de metadatos más común en las cámaras digitales [Bae10]. La especificación EXIF incluye cientos de etiquetas, entre las que se encuentran *marca* y *modelo*. Desafortunadamente, el seguimiento del estándar no es preceptivo.

Las técnicas basadas en el análisis de los metadatos de la imagen son las más sencillas y existen gran cantidad de trabajos enfocados en los diferentes tipos de metadatos tanto para la búsqueda de información como para la clasificación de imágenes e identificación de la fuente [BL04, BL05, Tes05, RCC+08, Are11].

Sin embargo, estas técnicas dependen en gran medida de los metadatos que los fabricantes deciden insertar cuando la imagen es generada. Asimismo, este método es el más vulnerable a modificaciones malintencionadas e incluso a la eliminación total de los metadatos ya sea intencionalmente o de manera inconsciente. Ejemplo de ello son algu-

nos programas de edición fotográfica que al editar o comprimir una imagen actualizan incorrectamente los metadatos o provocan la pérdida de los mismos.

A pesar de las debilidades de este tipo de técnicas, si existe el archivo de metadatos y de alguna manera se logra comprobar que no ha sufrido modificaciones externas, su uso es de gran utilidad para los analistas forenses, ya que del contenido de la imagen no se puede inferir toda la información contenida en los metadatos como es el caso de la información de GPS.

Un ejemplo del aprovechamiento de la información contenida en los metadatos es [Pla00] donde las etiquetas de tiempo han sido utilizadas satisfactoriamente para agrupar imágenes por eventos.

Un ejemplo más de la utilidad de los metadatos se presenta en [BL05] que mejora el proceso de clasificación de escenarios de imágenes con el apoyo del análisis de los metadatos. En el citado trabajo se presenta un método probabilístico para la fusión de las evidencias de los metadatos con la información proveniente de un clasificador del contenido de la imagen.

En los experimentos se consideran tres problemas para la clasificación de imágenes: interiores y exteriores, escenas de la naturaleza y de objetos creados por el hombre (esto es, naturales y artificiales) y, por último, la detección de puestas de sol. El análisis de las estadísticas de los metadatos de cada una de estas clases revela que algunas etiquetas como el tiempo de exposición, el flash y la distancia de los objetos son las más representativas para cada problema.

3.1.2 Técnicas Basadas en la Aberración de las Lentes

Durante el proceso de generación de la imagen en la parte del sistema de lentes se pueden introducir aberraciones. Existen diferentes tipos de aberraciones: esférica, coma, astigmatismo, curvatura de campo, distorsión radial y distorsión cromática. La distorsión radial es la que más consecuencias tiene sobre la imagen, especialmente en las cámaras que usan lentes baratas de gran angular (*wide angle*). La mayoría de cámaras digitales usan este tipo de lentes por cuestiones de coste.

En [Cho06] se propone la distorsión radial de la lente como la mejor técnica para la identificación de la fuente. La distorsión radial produce que las líneas rectas aparezcan como curvas en la imagen. Los autores concluyen que los diferentes fabricantes emplean diseños diferentes en los sistemas de lentes para compensar este efecto, dando como resultado que cada modelo de cámara exprese un único patrón de distorsión radial que ayuda a identificarla de manera única. El grado de distorsión radial de cada imagen se puede medir mediante un procedimiento que consta de tres fases: Detección de bordes, extracción de segmentos distorsionados y medición del error de la distorsión. En los experimentos se utilizaron tres cámaras diferentes y obtuvieron como resultado una precisión del 91,28 % en la identificación de la fuente.

3.1.3 Técnicas Basadas en la Interpolación de la Matriz CFA

Algunos autores consideran que la elección de la matriz de colores CFA y la especificación de los algoritmos de interpolación cromática generan algunas de las diferencias más marcadas entre los diferentes modelos de cámaras [BSM06, CAS+06, LH06, BSM08].

Como se ha comentado en la sección **??**, en las cámaras comerciales que tienen un solo sensor en lugar de tener sensores separados para cada componente del color

es crucial utilizar la matriz CFA y los algoritmos de interpolación cromática para capturar correctamente los detalles de la imagen. Estos algoritmos tienen un gran impacto en la calidad de los colores y en los contornos de la imagen resultante. En esencia, la interpolación cromática introduce un tipo específico de correlación entre los valores de colores de los píxeles de la imagen. La forma específica de estas dependencias (de estas "huellas dactilares") se puede extraer de las imágenes para diferenciar los algoritmos de interpolación cromática y así determinar marca y modelo de la cámara que generó una imagen.

Dentro de este tipo de técnicas se pueden diferenciar tres grupos:

- **Huellas en la Interpolación del Color**: En [BSM08] se presenta un algoritmo para identificar y clasificar las operaciones de interpolación cromática. La propuesta se basa en dos métodos para realizar el proceso de clasificación: el primer método utiliza un algoritmo *Expectation-Maximization* (EM) para analizar la correlación del valor de cada píxel con los valores de sus vecinos; el segundo método realiza un análisis de las diferencias entre píxeles (*inter-pixel*). Los experimentos se realizaron en dos fases: la primera fase tenía como objetivo evaluar la precisión del método de identificación de marca y modelo; la segunda evaluaba la precisión del método de identificación individual de la cámara cuando estas eran de la misma marca y modelo. Los resultados obtenidos en la identificación de la fuente de una imagen entre cuatro y cinco modelos diferentes tuvieron una precisión del 88 % y 84,8 % respectivamente. En los experimentos se utilizaron imágenes con ajustes automáticos y con el más alto nivel de calidad de compresión.

- **Modelo de Correlación Cuadrática de Píxeles**: En [LH06] se utilizan las correlaciones entre píxeles en el proceso de identificación de la fuente. Definen un modelo de correlación cuadrática de los pixeles y obtienen una matriz de coeficientes para cada banda de color. Para la clasificación utilizan redes neuronales. Se probó el método para cuatro cámaras con imágenes de dibujos animados y el éxito fue de un 95 % para una cámara, del 98 % para dos cámaras y del 100 % para el resto de las cámaras. También se realizaron pruebas para imágenes modificadas (incluyendo compresión) con resultados de un 80 % de éxito para una compresión JPEG del 80 %. Dado que las cámaras del mismo fabricante utilizan el mismo algoritmo de interpolación cromática, esta técnica no es eficiente entre distintos modelos del mismo fabricante. Asimismo, como demuestran los experimentos, no obtienen buenos resultados cuando las imágenes han sido modificadas o comprimidas.

- **Medidas de Similitud Binarias**: En [CAS+06] se utiliza un conjunto de medidas de similitud binarias como métricas para estimar la semejanza entre los planos de bits de una imagen. El supuesto fundamental de este trabajo es que el algoritmo de interpolación CFA de cada fabricante deja correlaciones a lo largo de los planos de bits de una imagen y pueden ser representados por este conjunto de medidas. En este estudio se utilizaron 108 medidas de similitud binarias que se obtienen para el propósito de la clasificación de imágenes.

  Los experimentos realizados con la técnica de medidas de similitud binaria para clasificar 3 grupos de cámaras obtuvieron un porcentaje de éxito entre el 81 % y el 98 %, mientras que para un grupo de 9 cámaras la precisión descendió al 62 % recolectando 200 imágenes de cada una de las cámaras. Las fotografías utilizadas en los experimentos fueron tomadas con las siguientes configuraciones: máxima resolución con un tamaño de 640 x 480 píxeles, a la luz del día y modo de enfoque automático. Claramente se puede apreciar que los resultados del método dependen del número de cámaras utilizadas en los experimentos.

3.1.4 Técnicas Basadas en las Características de las Imágenes

Estas técnicas utilizan un conjunto de características extraídas del contenido de la imagen para hacer la identificación de la fuente. Estas características se dividen en tres grupos: características de color, métricas de calidad de la imagen *Image Quality Metrics* (IQM) y estadísticas del dominio *wavelet*.

En [TLL07] se propone un método de identificación de la fuente utilizando las siguientes características: color, calidad de la imagen y dominio de la frecuencia. En el estudio adoptan la transformada *wavelet* como método para calcular las estadísticas del dominio *wavelet* y utilizan SVM [HCL03] para la clasificación. En los experimentos realizados se usaron cámaras digitales y dispositivos móviles. Los resultados obtenidos para cuatro modelos diferentes de dos fabricantes dieron una precisión cercana al 92 %.

En [MSGW08] se extiende la identificación de la fuente a diferentes dispositivos tales como teléfonos móviles con cámara integrada, cámaras digitales, escáneres y computadoras. Para ello se identifican en primer lugar las fuentes de variación entre los diferentes tipos de dispositivos y, posteriormente, entre diferentes modelos de los mismos.

En esta propuesta se usan las diferencias en el proceso de adquisición de la imagen de los dispositivos para formar dos grupos de características: coeficientes de interpolación de color y características de ruido. En los experimentos se utilizaron cinco modelos de teléfonos móviles, cinco modelos de cámaras digitales y cuatro modelos de escáneres para identificar el tipo de fuente. En los resultados globales se obtuvo un 93,75 % de precisión. En el análisis de identificación de marca y modelo de teléfonos móviles obtuvieron una precisión del 97,7 % para los cinco modelos.

En [MKY08] se propone un método que emplea las fases y las magnitudes de las estadísticas de bi-coherencia junto a las estadísticas de los coeficientes *wavelet*. Este método

captura las distorsiones únicas no lineales en el dominio *wavelet* producidas por las cámaras cuando realizan operaciones de procesamiento sobre las imágenes.

En primer lugar, para obtener la caracterización de las distorsiones no lineales se extraen las características de bi-coherencia: el bi-espectro de las señales se calcula dividiendo la señal en N segmentos (posiblemente traslapados). Posteriormente, se calcula la Transformada de Fourier para cada segmento, y se promedian las estimaciones individuales. La magnitud (la media de la magnitud de la bi-coherencia) y la fase (la entropía negativa de la fase de la bi-coherencia) se calculan como características estadísticas. Para reducir la memoria y la sobrecarga computacional implicada en el cálculo total de la bi-coherencia de cuatro dimensiones de las imágenes, restringen su análisis a filas, columnas y rodajas radiales a través del centro de la imagen de una sola dimensión.

Es interesante observar que en este trabajo no hay restricciones rigurosas en cuanto a la selección de las imágenes de muestra, debido a que con la aplicación de las estadísticas de bi-coherencia no es necesario extraer la información asociada con el contenido de la imagen (por ejemplo, segmentos de línea). A continuación, se emplea la descomposición *wavelet* en cuatro niveles para dividir el espacio de la frecuencia en cuatro escalas y orientaciones. Después se calculan cuatro estadísticas (media, varianza, asimetría y curtosis) de cada coeficiente de la sub-banda así como los errores de predicción lineal para cada orientación, nivel y canal de color.

Estas estadísticas componen el segundo grupo de vectores de características estadísticas utilizadas para la identificación de la cámara fuente. Una vez que las estadísticas de bi-coherencia y *wavelet* se calculan, el algoritmo *Sequential Forward Featured Selection* (SFFS) [PNK94] se utiliza para reducir la correlación entre las características y la carga computacional manteniendo la misma precisión de la clasificación. El método SFFS analiza todas las características y construye el conjunto más representativo de ellas, añadiendo y quitando características hasta que no haya más mejoras disponibles.

Por último, las características más representativas son clasificadas por una SVM utilizando un *kernel Radial Basis Function* (RBF). Se realizaron experimentos bajo las siguientes condiciones: 6 modelos de cámara de 4 fabricantes, imágenes de diferentes resoluciones, formato JPEG, un total de 2.100 (350 de cada cámara) imágenes de tomas típicas variando la naturaleza de las escenas. Como resultado obtuvieron un notable porcentaje de precisión en la identificación de la fuente que supera el 97 % distinguiendo diferentes modelos del mismo fabricante. Caben posibles mejoras mediante la incorporación de otras características como las utilizadas en [WGKM09].

En [WGKM09] se propone un método para la identificación de la cámara fuente mediante la extracción y clasificación de las estadísticas de las características *wavelets*. Este método está compuesto por tres fases: extracción, selección y clasificación de características *wavelet*. Las características sobresalientes de dominio *wavelet* se extraen para integrar un modelo estadístico de imagen a partir de los coeficientes *wavelet*, incluyendo 216 características *wavelet* de primer orden y 135 características de co-ocurrencia de segundo orden. En este estudio las características del dominio *wavelet* se consideran más representativas

y son preferidas a las características espaciales (color de la imagen e IQM) y matrices de filtros de color CFA. De manera análoga a [MKY08] se realiza una descomposición *wavelet* en 4 niveles basada en *Separable Quadrate Mirror Filters* (QMF) para dividir el espacio de la frecuencia, se extraen las mismas cuatro estadísticas (media, varianza, asimetría y curtosis) junto a los errores de predicción lineal. Ya que estas cuatro estadísticas no brindan información sobre la correlación de la textura se usan las características de co-ocurrencia para la extracción de características de textura de la imagen ya que según [RH99] son las más adecuadas para este propósito. A partir de las características de co-ocurrencia se extraen las características de segundo orden (energía, entropía, contraste, homogeneidad y correlación).

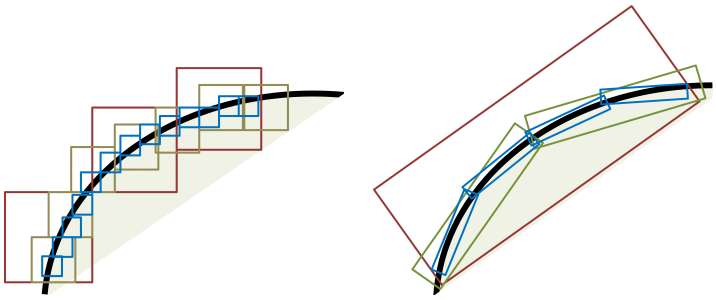
Por último, y al igual que en [MKY08], se seleccionan las características más representativas utilizando un algoritmo SFFS y se clasifican utilizando una SVM con un *kernel* no lineal. Bajo las mismas condiciones que en los experimentos realizados en [MKY08] logran distinguir entre diferentes modelos del mismo fabricante de cámaras Canon.

Este trabajo puede mejorarse evaluando la robustez del sistema de identificación propuesto por el vector de características, así como ampliando la base de datos de imágenes para realizar las pruebas incluyendo dispositivos móviles, además de cubrir más marcas, modelos, texturas y contenidos.

En [OA11] se plantea una técnica para diferenciar imágenes usando las transformaciones de la familia *wavelet*. Proponen modelos estadísticos para *ridgelet* y sub-bandas *contourlet*, que se describen seguidamente.

- **La Transformada Ridgelet**: Básicamente, la Transformada *Wavelet* es buena para la representación de singularidades de cero dimensiones o puntos. Sin embargo, las señales de 2-D (imágenes) por lo general contienen singularidades 1-D (bordes y esquinas). Para resolver la debilidad de la Transformada *Wavelet* en dos o más dimensiones en [CD99] se desarrolló un nuevo sistema de representaciones llamado *ridgelet* que puede cubrir eficazmente las singularidades de líneas en dos dimensiones. La idea es utilizar la Transformada Radón o *Radon Transform* (RAT) para mapear las singularidades de línea a singularidades punto, y así luego poder manejarlas de una manera más efectiva mediante el uso de la Transformada *Wavelet*.

- **Transformada Contourlet**: En la pintura se utilizan líneas y contornos en lugar de puntos para crear imágenes. La representación *wavelet* de una imagen es el equivalente a usar puntos en lugar de líneas; en este caso la imagen no es clara y su construcción es más difícil, como se puede observar en la Figura 3.1(a). Del mismo modo, la representación llamada *contourlet* [DV05] es el equivalente a usar líneas, lo que simplifica la construcción de la imagen y le da un aspecto más realista como se aprecia en la Figura 3.1(b).



(a) *Representación Wavelet* (b) *Representación Contourlet*

Figura 3.1: Representación *wavelet* vs representación *contourlet*

De acuerdo a los resultados de estudios previos [DV05], la representación eficiente de una imagen debería satisfacer las siguientes características:

- **Resolución Múltiple**: La representación debe ser una aproximación satisfactoria de la imagen, teniendo en cuenta las resoluciones altas y bajas.
- **Localización**: Los elementos básicos deberán estar localizados en ambos dominios, tanto el espacial como el espectral (de frecuencia).
- **El Muestreo Crítico (*Critical Sampling*)**: La representación debe formar un marco o una base con bajo nivel de redundancia.
- **Direccionalidad**: Una buena representación debe tener elementos base en diferentes direcciones.
- **Anisotropía**: Para capturar contornos suaves en las imágenes, la representación debe contener elementos base usando una variedad de formas alargadas con diferentes relaciones de aspecto.

Las transformadas *wavelet* cubren las primeras tres propiedades, las transformadas *ridgelet* las primeras cuatro, y las transformadas *contourlet* las cinco, esto es, cubren todas las propiedades.

Después de definir los modelos estadísticos para los coeficientes *ridgelet* y contourlet, se realiza la extracción de características. Para cada sub-banda de la transformada *wavelet* se calculan ocho características estadísticas a partir de coeficientes, así como la predicción de error entre los coeficientes mediante el uso de los modelos estadísticos propuestos.

Por último, se aplica un algoritmo *Sequential Floating Search* (SFS) para selección de características y una SVM para la clasificación.

El método basado en *wavelets* considera 216 características útiles sólo para la representación de una dimensión, el enfoque basado en *ridgelets* toma 48 características, y la aproximación de *contourlets* contempla un total de 768 características.

La mejora de los resultados aplicando tanto las transformaciones *ridgelet* como las transformaciones *contourlet* es razonable debido al hecho de que se cuenta con las estadísticas de más de tres direcciones, teniendo en cuenta las cinco propiedades de una representación de imagen eficiente.

Los *contourlets* y *ridgelets* no sólo son efectivos para diferenciar entre modelos de cámaras, sino también para diferenciar entre imágenes producidas por diferentes cámaras o escáneres del mismo modelo. De cualquier manera los autores consideran que podrían implementar mejoras experimentando con diferentes algoritmos de selección como es el caso de SFFS.

Los estudios basados en las técnicas *wavelet* tienen buenos resultados. Sin embargo, los experimentos que se han realizado se enfocan a cámaras digitales tradicionales, dejando a un lado los dispositivos móviles que es uno de los campos que está ganando más terreno cada día como ya se ha mencionado anteriormente. En el Anexo A se describen algunas generalidades de la transformada *wavelet*.

En [LLC+12] se propone un método que emplea la densidad marginal de los coeficientes de la Transformada Coseno Discreta o *Discrete Cosine Transform* (DCT) en las coordenadas de frecuencia baja y las características de densidad del vecindario (*neighbouring joint density*) en el domino DCT. Adicionalmente, se utiliza la agrupación jerárquica (*hierarchical clustering*) y una máquina de soporte vectorial SVM con *kernel* RBF lineal para detectar los teléfonos inteligentes fuente y los procesamientos aplicados a las imágenes. En los experimentos realizados con imágenes de diferentes factores de escala pertenecientes a cinco modelos de teléfonos inteligentes de cuatro fabricantes se obtuvo entre el 86,36 % y el 99,91 % de exactitud, alcanzando mejores resultados con un *kernel* lineal. A pesar de los resultados satisfactorios, esta propuesta puede mejorarse mediante la optimización de los parámetros del *kernel*, el aumento el tamaño del conjunto de imágenes de prueba y la adopción de un algoritmo sofisticado para la selección de las características.

3.1.5 Técnicas Basadas en el Uso de las Imperfecciones del Sensor

Estas técnicas se basan en el estudio de las huellas que los defectos del sensor descritos en la sección 2.1.3.2 pueden dejar sobre las imágenes. Estas técnicas se dividen en dos ramas: defectos de píxel y patrón de ruido del sensor *Sensor Pattern Noise* (SPN). En la primera se estudian los defectos de píxel, los píxeles calientes, los píxeles muertos, los defectos de fila o columna, y los defectos de grupo. En la segunda se construye un patrón del ruido promediando los múltiples residuos de ruido obtenidos mediante algún filtro de eliminación de ruido. La presencia del patrón se determina utilizando algún método de clasificación como correlación o máquinas SVM.

En [GBK+01] se estudian los defectos de los píxeles en los sensores de tipo CCD, centrándose en la evaluación de diferentes características para examinar las imágenes e identificar la fuente: defectos del sensor CCD, formato de los archivos usados, ruido introducido en la imagen y marcas de agua introducidas por el fabricante de la cámara.

Entre los defectos del sensor CCD considerados se encuentran los puntos calientes, los píxeles muertos, los defectos en grupo y los defectos de fila o columna. En sus resultados se observa que cada una de las cámaras tiene un patrón de defecto diferente. Sin embargo, también se señala que el número de defectos en los píxeles para una cámara es diferente entre fotos y varía demasiado en función del contenido de la imagen. Asimismo, se revela que el número de defectos cambia con la temperatura. Por último, el estudio encontró que las cámaras con CCD de alta calidad no tienen este tipo de problema. También es cierto que la mayoría de las cámaras tienen mecanismos adicionales para compensar este tipo de problemas. Al considerar únicamente los defectos de los sensores de tipo CCD este estudio no es aplicable al análisis de imágenes generadas por dispositivos móviles.

En [LFG06] se analiza el patrón de ruido del sensor de un conjunto de cámaras, el cual funciona como una huella dactilar, permitiendo la identificación única de cada cámara. Para obtener este patrón se realiza un promedio del ruido obtenido a partir de diferentes imágenes utilizando un filtro de eliminación de ruido. Para identificar la cámara a partir de una imagen dada, se considera el patrón de referencia como una marca de agua cuya presencia en la imagen es establecida mediante un detector de correlación. El estudio se realizó con 320 imágenes procedentes de 9 modelos distintos de cámaras. También se demuestra que este método está afectado por algoritmos de procesamiento de la imagen como la compresión JPEG y la corrección *gamma*. Los resultados para fotografías con diferentes tamaños y recortadas no son satisfactorios [VCEK07].

En [CESR12] se propone un enfoque para la identificación de la cámara fuente considerando escenarios abiertos, donde a diferencia de los escenarios cerrados no se da por sentado contar con acceso a todas las posibles cámaras de origen de la imagen. Esta propuesta comprende tres fases: definición de las regiones de interés, determinación de las características e identificación de la cámara fuente. Las diferentes regiones de las imágenes pueden contener información distinta sobre la huella digital de la cámara fuente. Este enfoque, en contraste con otros, considera diferentes áreas de interés *Region Of Interest* (ROI) y no sólo la región central de la imagen. Para cada imagen se definen nueve ROIs como se puede ver en la Figura 3.2.

Se asume que estas regiones coinciden con el eje principal de la lente y, por lo tanto, deben tener más detalles de la escena porque los fotógrafos aficionados por lo general centran el objeto de interés en el centro de la lente. Además, las regiones de la 6 a la 9 proporcionan información importante debido a que las simetrías del sistema de lentes de algunas cámaras generan un efecto de bordes oscuros en las fotografías, lo que implica una caída de la intensidad radial desde el centro de la imagen provocando una pérdida de brillo o saturación en la periferia.

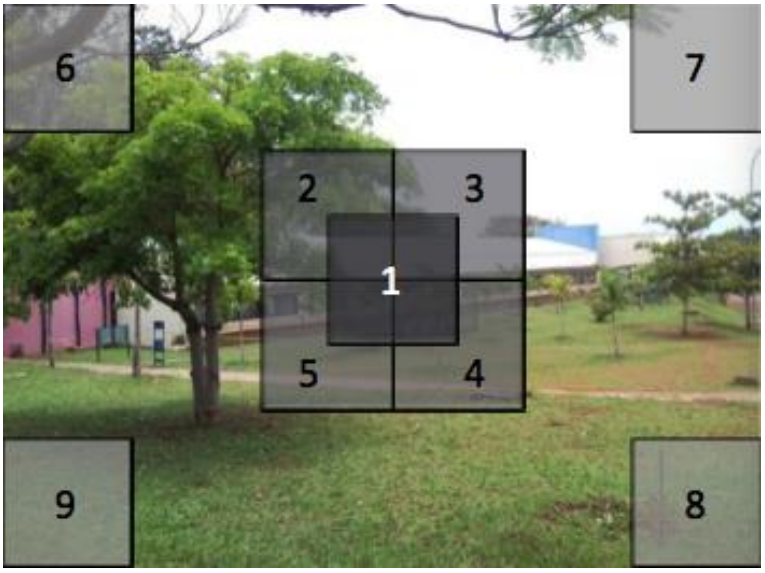


Figura 3.2: Regiones de interés

El uso de las regiones de interés permite trabajar con imágenes de diferentes resoluciones sin la necesidad de rellenar con ceros las imágenes y sin el uso de artefactos de interpolación de color. Para determinar las características se calcula el SPN para cada uno de los canales R, G y B. Asimismo, se calcula el SPN para el canal Y (luminancia), que es una combinación de los tres canales *Red-Green-Blue* (RGB) (como una versión en escala de grises de la imagen), generándose un total de 36 características para representar cada imagen. Después, las imágenes tomadas por la cámara bajo investigación son etiquetadas como la clase positiva y las tomadas por las cámaras disponibles restantes como las clases negativas. Después de la fase de entrenamiento de la SVM en la que se calcula el hiper-plano que separa los casos positivos y negativos toman en cuenta las clases desconocidas del escenario abierto moviendo el hiper-plano generado por un valor dado ya sea hacia adentro (hacia las clases positivas) o hacia afuera (las clases negativas).

Mediante el movimiento del hiper-plano se puede variar que tan estricto se desea ser para determinar si una imagen pertenece a una clase o no. A este proceso se le denomina modelado de límites de decisión o *Decision Boundary Carving* (DBC). En los experimentos se utiliza un conjunto de 25 cámaras digitales de 9 fabricantes, 150 imágenes en formato JPEG de cada cámara con diferentes configuraciones de luz, *zoom* y *flash*. Los resultados de los experimentos mostraron una precisión del 94,49 %, del 96,77 % y del 98,10 %, utilizando conjuntos abiertos con 2/25, 5/25, y 15/25 cámaras, respectivamente, definiendo un conjunto abierto x/y como el conjunto de y cámaras donde x cámaras son usadas para entrenar y probar las imágenes que pueden pertenecer a cualquiera de las cámaras x conocidas, así como a las otras y-x cámaras desconocidas.

3.1.6 Resumen

En la Tabla 3.1 se muestran los resultados de la evaluación de las diferentes técnicas de identificación de la cámara fuente. La información que no se detallaba en los artículos correspondientes ha sido cumplimentada con las letras ND (No Detallado).

Tabla 3.1: Comparativa sobre las diferentes técnicas de identificación de la cámara fuente

Grupo Técnica	**Características de la Imagen**			**Interpolación de la Matriz CFA**		**Imperfecciones del Sensor**			**Transformación Wavelet**		
Técnica	[TLL07]	[MSGW08]	[LLC+12]	[CAS+06]	[BSM08]	[GBK+01]	[LFG06]	[CESR12]	[MKY08]	[WGKM09]	[OA11]
Clasificador	SVM	SVM	SVM	SVM	SVM	ND	ND	SVM	SVM	SVM	SVM
Núcleo SVM	Lineal	Lineal	Lineal y No-lineal RBF	Lineal y No-lineal RBF	Lineal y No-lineal RBF	ND	ND	No-lineal RBF	No-lineal RBF	No-lineal RBF	No-lineal RBF
Número de fabricantes	2	5	4	3	5	1	5	9	4	4	3
Número de modelos	4	5	5	9	2	2	9	25	6	6	3
Número de imágenes por cámara	150 Entrenar: 60 Probar: 90	100 Entrenar: 90 Probar: 10	599	200	600	ND	320	50	350 Entrenar: 100 Probar: 150	350 Entrenar: 100 Probar: 150	2000 Entrenar: 1000 Probar: 1000
Resoluciones	1600 x 1200	ND	Diferentes	Diferentes	Diferentes	640 x 480	Diferentes	Diferentes	Diferentes	Diferentes	ND
Formato	JPEG	JPEG	JPEG	ND	JPEG	ND	JPEG	JPEG	JPEG	JPEG	ND
Aplicado a móviles	Sí	Sí	Sí	Sí	No	No	No	Sólo 2 entre 25 cámaras	No	No	No
Aplicado a diferentes modelos del mismo fabricante	Sí	No	Sí	Sí	Sí	ND	Sí	Sí	Sí	Sí	No

3.2 Ataques al Análisis Forense de Imágenes

En comparación con el destacado papel de las imágenes digitales en la sociedad multimedia de hoy en día, la investigación en el campo de la autenticidad de la imagen se encuentra todavía en un estadio muy preliminar. La mayoría de las publicaciones en este campo emergente todavía carece de discusiones rigurosas y robustas contra los falsificadores estratégicos, que prevén la existencia de técnicas forenses [GKWB07].

El área que se encarga de estudiar ataques a las técnicas de análisis forense de imágenes es conocida como *counter-forensics*. Los ataques contra los algoritmos forenses de imágenes digitales son aquellas técnicas cuyo objetivo es confundir sistemáticamente a los procedimientos de identificación de la fuente de la imagen o de detección de manipulaciones maliciosas en las imágenes. Estos ataques pueden tener uno de los siguientes objetivos:

1. Camuflaje de post-procesamientos maliciosos sobre la imagen.
2. Destrucción de la identificación correcta del origen de la imagen.
3. Falsificación del origen de imagen.

El análisis forense de imágenes digitales se ha convertido en un tema de interés en los últimos años. En sus inicios la parte académica encontró utilidad del análisis forense de imágenes en ámbitos como aplicación de la ley, inteligencia, investigaciones privadas y medios de comunicación. El análisis forense surge con la idea de restablecer la confiabilidad en las imágenes digitales que de otro modo se consideraban muy fácilmente modificables. Como en la mayoría de campos de estudio existe una contracorriente, en este caso, personas como espías o estafadores hacen esfuerzos para manipular las imágenes en su propio beneficio usando el conocimiento del análisis forense de imágenes para borrar o incluso suplantar las huellas o rastros que se utilizan para determinar la identidad de las imágenes. Muchos de los algoritmos forenses existentes en la literatura no fueron diseñados teniendo en cuenta ese tipo de comportamiento y como consecuencia son fáciles de engañar.

La posibilidad de copiar las huellas digitales de una imagen se puede convertir en un ciclo infinito que puede permitir que personas inocentes sean inculpadas, también que criminales aseguren que las pruebas son resultados de una falsificación. Al final, la confianza en las técnicas forenses de imágenes se podría ver comprometida. Es por esto que surge la necesidad considerar los posibles ataques en el momento de diseñar técnicas de análisis forense en imágenes digitales.

Así como en el área de seguridad el estudio de los ataques permite mejorarla, los métodos forenses de imágenes se pueden beneficiar del estudio de las técnicas de ataque para robustecer los algoritmos de las próximas generaciones.

3.2.1 El Camuflaje de Post-Procesamientos

Estas técnicas tienen como objetivo ocultar la existencia de algún proceso aplicado a una imagen analizando los rasgos que éstos dejan sobre la imagen durante su aplicación

para así poder contrarrestarlos.

Entre las investigaciones realizadas sobre los rasgos de los algoritmos en las imágenes se encuentran el estudio de las dependencias introducidas durante el re-dimensionamiento o la rotación de las imágenes [PF05], el estudio de los coeficientes estadísticos de los JPEG para detectar la re-compresión [LF03], y el análisis de la fase de congruencia para detectar la composición de imágenes a través del recortado y pegado de diferentes imágenes [CSS07].

Para ejemplificar este tipo de técnicas se describe a continuación la propuesta presentada en [GKWB07] para ocultar el proceso de re-muestreo (*resampling*).

El re-muestreo es el redimensionamiento con interpolación de las imágenes. Este proceso es muy común en las operaciones primitivas de imágenes como escalamiento y rotación.

Los algoritmos detectores de re-muestreo se basan en la búsqueda de las dependencias sistemáticas y periódicas entre píxeles vecinos insertadas cuando se aplica la operación de re-muestreo. Esta periodicidad se debe a la matriz de muestreo equidistante utilizada para realizar dicha operación [PF05].

Para ocultar el re-muestreo es necesario romper las equidistancias periódicas introduciendo distorsiones geométricas conocidas como ataques de marca de agua. En este caso se superpone un vector de distorsión aleatoria a las posiciones de cada píxel donde un parámetro determina el grado de distorsión introducido. Para evitar generar características visibles en la imagen como ruido se debe modular la fuerza de la distorsión empleando dos detectores de bordes: uno en dirección vertical y otro en dirección horizontal.

El ataque para generar una imagen $\tilde{\mathbf{y}}$ a partir de la imagen $\mathbf{x}$ aplicando el re-muestreo sin dejar rastro de dicha operación consiste en los siguientes pasos:

1. *Calcular el componente de baja frecuencia: A la imagen $\boldsymbol{x}$ de entrada se le aplica el re-muestreo y a este resultado se le aplica el filtro de la mediana.*

2. *Calcular el componente de alta frecuencia: Restar a la imagen $\boldsymbol{x}$ el resultado de aplicarle el filtro de la mediana, aplicar a este resultado el re-muestreo con una distorsión geométrica y una modulación de los bordes; la información de los bordes es extraída de la imagen con muestro aplicado del paso 1 antes de aplicar el filtro de la mediana.*

3. *Obtener la imagen final $\tilde{\boldsymbol{y}}$ sumando los resultados del paso 1 y 2.*

En la Figura 3.3 se puede observar el diagrama de bloques del ataque.

Los resultados que obtuvieron en los experimentos realizados con este ataque apuntan a que es una propuesta prometedora ya que tiene una tasa de falsos positivos *False Acceptance Rate* (FAR) inferior al 1 %.

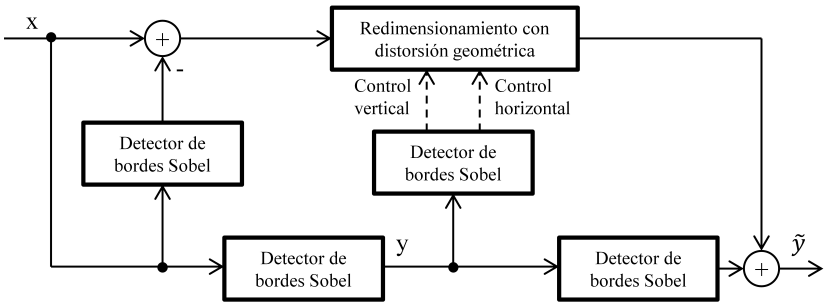


Figura 3.3: Diagrama de bloques enfoque de doble vía para ocultar re-muestreo

3.2.2 Manipulación de la Identificación de la Fuente

Así como para el proceso de identificación de la fuente se usa la extracción del ruido del sensor en la imagen, un contraataque lógico para esta técnica consta en la eliminación del ruido del sensor. Dando un paso más adelante se puede pensar también en la posibilidad de eliminar el ruido del sensor de la imagen y sustituirlo por el ruido del sensor que le pertenezca a otra cámara.

3.2.2.1 Destrucción de la Identidad de una Imagen

En [GKWB07] se demostró que la resta de las características del dominio *wavelet* de las imágenes no es suficiente para eliminar el ruido de una imagen, además de que este procedimiento deja rastros visibles sobre la imagen. Existe otro método bastante conocido para la eliminación del ruido de una imagen llamado corrección de sensibilidad o *flatfielding*. Este método es usado típicamente en astronomía o en el proceso de escaneado de planos para mejorar la calidad de las imágenes.

La corrección de sensibilidad se realiza en base a los principales componentes del ruido de la imagen: el ruido de patrón fijo FPN y el ruido de respuesta no uniforme PRNU.

El ruido FPN se calcula con la ecuación 3.1 en términos de un marco oscuro d promediando K imágenes x_{oscura} capturadas en un ambiente completamente oscuro que se puede emular cubriendo completamente el lente de la cámara:

$$d = \frac{1}{K} \sum x_{oscura} \tag{3.1}$$

El ruido PRNU se calcula con la ecuación 3.2 en términos de un marco plano (*flatfield*) f promediando L imágenes $x_{iluminada}$ de una escena iluminada homogéneamente. A las L imágenes se les elimina el ruido FPN mediante la resta del marco oscuro d antes de promediarlas.

$$f = \frac{1}{L} \sum_{L} (x_{iluminada} - d) \tag{3.2}$$

Como se describe en [LFG06, GKWB07], los atacantes pueden intentar evitar la identificación correcta de la fuente ya que existe la posibilidad de eliminar y extraer la huella de una imagen. La destrucción de la huella de una imagen x generada con una cámara específica se realiza con la ecuación 3.3 restando a la imagen original x el marco oscuro d y dividiendo el resultado de la resta entre el marco plano f.

$$\tilde{x} = \frac{x - d}{f} \tag{3.3}$$

A pesar que los resultados obtenidos con esta técnica son buenos, se presentan algunos inconvenientes:

- Llevar a cabo una corrección de sensibilidades perfecta en un gran número de fotos es difícil ya que los parámetros para calcular el PRNU y el FPN deben coincidir con los de la imagen a atacar.

- En la propuesta se asume que el atacante puede tener acceso a la cámara fuente de la imagen x para generar los marcos oscuros y planos y éste no es un escenario próximo a la realidad.

Existen otras posibilidades menos robustas para destruir la identidad que en ciertos casos podrían ser efectivas ya que no necesitan contar con imágenes procedentes de la cámara origen para generar el marco oscuro y el marco plano, pero a cambio de esta facilidad la calidad de la imagen puede verse reducida y podrían introducirse algunos rasgos visuales. Por ejemplo, es posible rotar la imagen unos pocos grados, escalar la imagen, o aplicar un filtro de desenfoque gaussiano.

3.2.2.2 Falsificación de la Identidad de una Imagen

De igual forma que se puede eliminar el ruido en una imagen haciendo uso de la técnica de corrección de sensibilidad, se puede inyectar el ruido de la imagen de otra cámara diferente mediante la corrección de sensibilidad inversa con la ecuación 3.4 [GKWB07].

$$\tilde{y} = \tilde{x} \cdot f_{falsa} + d_{falsa} \tag{3.4}$$

Donde, f_{falsa} y d_{falsa} corresponden a la cámara que se pretende plagiar y $\tilde{x}$ es la imagen original sin ruido.

En [SLFK10] se propone un algoritmo para falsificar la identidad de una cámara, a continuación se describen los pasos a seguir:

1. *Calcular el promedio de las huellas F(**C1**) de la cámara **C1** con la que se atacará..*

2. *Tomar una fotografía **P** con la segunda cámara **C2**.*

3. *Sumar F(**C1**) a la fotografía **P**.*

En el caso de que las dimensiones de *F(**C1**)* y ***P*** no coincidan, es necesario aplicar un recorte o una reconstrucción para igualar el tamaño de las imágenes.

También se propone una mejora al algoritmo de falsificación anterior para enmascarar los rasgos de la cámara **C2**. Esta técnica se presenta en el siguiente algoritmo.

1. *Calcular el promedio de las huellas F(**C1**) de la cámara **C1** con la que se atacará.*
2. *Calcular el promedio de las huellas F(**C2**) de la cámara **C2**.*
3. *Tomar una fotografía **P** con la cámara **C2**.*
4. *Restar F(**C2**) a **P**.*
5. *Sumar F(**C1**) a la fotografía **P**.*

Al restar *F(**C2**)* se trata de eliminar la correlación entre la fotografía ***P*** y la cámara ***C2***.

3.2.3 Detección de Falsificación de la Identidad de una Imagen

Una vez estudiada la técnica de falsificación de la huella de una imagen y la inyección de ésta en otra imagen, surge la pregunta de si es posible detectar cuándo se ha sufrido un ataque de este tipo. La respuesta a esta pregunta es sí, y se puede lograr mediante el análisis de las diferencias entre las propiedades de un patrón de ruido copiado [SLFK10, GFC11].

Para explicar la detección de la falsificación de la identidad de una imagen se presenta el siguiente escenario de ataque:

1. ***Alicia**, la víctima, sube algunas fotografías adquiridas con su cámara **C** a una red social en internet.*
2. ***Eva**, la atacante, obtiene **N** de esas fotografías y calcula la huella K'_{E} perteneciente a la cámara de **Alicia**.*
3. ***Eva** implanta la huella K'_{E} en una imagen **J** tomada con una cámara **C'** con el propósito de hacer parecer que **Alicia** fue la autora de esa imagen falsificada **J'**.*

Entonces, surge la siguiente pregunta: ¿La imagen **J'** fue falsificada? Para resolver esta pregunta Alicia en su defensa puede hacer uso de su cámara **C** y un conjunto **F** de fotografías compuesto por las fotografías que Eva robó más algunas extras que le pertenezcan. El escenario de defensa de Alicia propuesto en [GFC11] sería:

1. *Calcular la huella de su cámara K'_{A} utilizando imágenes planas inocentes que no hayan sido manipuladas por **Eva** (para obtener una mejor estimación de la huella).*
2. *Calcular el ruido residual PRNU W_{J}, de la imagen **J'**.*

3. *Calcular el ruido residual PRNU* $W_{\boldsymbol{I}}$ *de una de las imágenes I utilizadas por* ***Eva*** *para realizar el ataque. Debido a que* $W_{\boldsymbol{I}}$ *participó en el cálculo de* $K'_{\boldsymbol{E}}$ *contiene una versión escalada del ruido residual total compartiendo rasgos característicos con* $W_{\boldsymbol{J'}}$.

4. ***Alicia*** *calcula las correlaciones:*

   - $C_{\mathbf{I,J'}}$=corr($W_{\mathbf{I}}$, $W_{\mathbf{J'}}$)
   - $C_{\mathbf{I,K'A}}$=corr($W_{\mathbf{I}}$, $K'_{\mathbf{A}}$)
   - $C_{\mathbf{J',K'A}}$=corr($W_{\mathbf{J'}}$,$K'_{\mathbf{A}}$)

5. *Evaluar si* ***J'*** *fue atacada realizando la prueba del triángulo como se muestra en la Figura 3.4 con las correlaciones calculadas en el paso 4.*

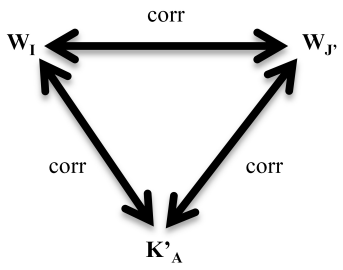


Figura 3.4: Correlaciones de la prueba del triángulo

Esta prueba se basa en el hecho de que el valor de correlación $C_{\mathbf{I,J'}}$ de las imágenes I que no fueron utilizadas para falsificar **J'** puede ser estimado de las correlaciones $C_{\mathbf{I,K'A}}$ y $C_{\mathbf{J',K'A}}$. En caso de que la imagen I haya sido utilizada para la falsificación la correlación $C_{\mathbf{I,J'}}$ tendría un valor mayor que $C_{\mathbf{I,K'A}}$ y $C_{\mathbf{J',K'A}}$.

Por supuesto este contraataque también es vulnerable ya que al poder distinguir entre las huellas originales y las que son copias los atacantes podrían comenzar a generar imágenes ilegales con características que se ajusten a las de las fotografías encontradas de la persona a la que se pretende atacar. Sin embargo, esta propuesta incrementa los esfuerzos requeridos para culpar a alguien inocente.

Capítulo 4

Contribuciones

El objetivo de este capítulo es presentar las contribuciones de este trabajo. Por un lado, se presenta un algoritmo para realizar la identificación de la marca y modelo del dispositivo móvil fuente de una imagen, describiendo cómo se realiza la extracción de la huella del sensor de una imagen de manera individual, la estimación del patrón del ruido del sensor cuando se cuenta con varias imágenes y la extracción de características requeridas para la clasificación y la identificación de la fuente. Por otro lado, se detalla un algoritmo para falsificar la identidad de una imagen.

4.1 Consideraciones Generales

Antes de nada, conviene reseñar que los algoritmos que se proponen tienen como ventaja el hecho de no requerir acceso a la cámara del dispositivo móvil fuente, siempre y cuando se cuente con imágenes procedentes del mismo.

Cada una de las técnicas estudiadas en el capítulo 3 para la identificación de la fuente tiene sus fortalezas y debilidades. Asimismo, unas son más apropiadas que otras en un escenario particular como se explica en la sección 2.1.4.

Las anteriores técnicas han sido realizadas en su mayoría teniendo en cuenta las características concretas de las DSCs y no de las cámaras incorporadas en dispositivos móviles.

En base al análisis realizado de las diferentes técnicas de identificación de la fuente se detectó que para el caso particular de los dispositivos móviles las técnicas más adecuadas son las basadas en el ruido del sensor por el tipo de sensor que usan y las basadas en las características *wavelet* por su efectividad. Estas dos técnicas son las que han obtenido mejores resultados. Sin embargo la mayoría de los trabajos realizados se han enfocado únicamente en cámaras tradicionales olvidándose de las cámaras de dispositivos móviles.

Como se mencionó en la sección 1.1, las cámaras de dispositivos móviles constituyen un área de estudio que hoy en día requiere atención. Algunos trabajos incluyen experimentos con cámaras de teléfonos móviles sin considerar otro tipo de dispositivos móviles utilizando además un número muy reducido de teléfonos móviles y la mayoría no considera diferentes modelos de la misma marca alejándose de escenarios reales (véase Tabla 3.1).

A la vista de las consideraciones anteriores, este trabajo propone mejorar la identificación de dispositivos móviles fuente combinando dos técnicas: las imperfecciones en el sensor y la transformada *wavelet*.

4.2 Algoritmo de Identificación de la Fuente

Para poder identificar la marca y el modelo de la cámara digital de un dispositivo móvil se requiere de un algoritmo que nos permita extraer la huella del sensor y otro que nos permita obtener las características de las huellas obtenidas para así poder clasificarlas e identificarlas.

4.2.1 Algoritmo de Extracción de la Huella y Patrón de la Huella del Sensor

A continuación se muestra el algoritmo que se debe seguir para extraer la huella del sensor, también conocida como ruido residual:

- Entrada: m imágenes de las que se extraerá el ruido del sensor.
- Salida: huella del sensor en caso de $m = 1$ o el patrón de la huella del sensor en caso de $m > 1$.

Para cada imagen de entrada I_n *donde* $n \in \{1, \ldots, m\}$

Paso 1. *Estimar la imagen sin ruido del sensor* $I_{n_{limpia}}$

Paso 1.1. *Realizar una descomposición wavelet de 4 niveles de* I_n

Para cada nivel de la descomposición wavelet

Para cada componente wavelet c *donde* $c \in \{H, V, D\}$*:*

Paso 1.1.1. *Calcular la varianza local:*

En caso de varianza adaptativa:

– *Calcular la varianza local para las ventanas de tamaño* $W \times W$ *para* $W \in \{3, 5, 7, 9\}$ *de vecindario* N *con:*

$$\hat{\sigma}^2(i,j) = \text{máx}\left(0, \frac{1}{W^2}\sum_{(i,j)\in N} c^2(i,j) - \sigma_0^2\right)$$

– *Tomar como valor final de la varianza la mínima de las cuatro varianzas estimadas en el paso anterior mediante:*

$$\hat{\sigma}^2(i,j) = \text{mín}\left(\sigma_3^2(i,j), \sigma_5^2(i,j), \sigma_7^2(i,j), \sigma_9^2(i,j)\right)$$

En caso de varianza no adaptativa:

– *Calcular la varianza local para una ventana de tamaño* $W \times W$ *para* $W = 3$ *de vecindario* N *con:*

$$\hat{\sigma}^2(i,j) = \text{máx}\left(0, \frac{1}{W^2}\sum_{(i,j)\in N} c^2(i,j) - \sigma_0^2\right)$$

Paso 1.1.2. *Aplicar el filtro de Wiener mediante:*

$$c_{limpio}(i,j) = \quad c(i,j)\frac{\hat{\sigma}^2(i,j)}{\hat{\sigma}^2(i,j)+\sigma_0^2}$$

Paso 1.2. *Aplicar la Transformada Inversa Wavelet con los componentes limpios de ruido calculados.*

Paso 2. *Calcular el ruido residual con:*

$$I_{n_{ruido}} = \quad I_n - I_{n_{limpia}}$$

Paso 3. *Aplicar a* $I_{n_{ruido}}$ *un promediado a cero.*

Paso 4. *Aumentar en* $I_{n_{ruido}}$ *el peso del canal verde:*

$$I_{n_{ruido}} = \quad 0.3 \cdot I_{n_{ruido_R}} + 0.6 \cdot I_{n_{ruido_G}} + 0.1 \cdot I_{n_{ruido_B}}$$

Paso 5. *Si* $m > 1$ *promediar los ruidos residuales para obtener el patrón del ruido mediante:*

$$P_{ruido} = \frac{1}{m}\sum_{i=1}^{m} I_{n_{ruido}}$$

El algoritmo de extracción de la huella propuesto se basa en las ideas principales de [LFG06]. A continuación se describen detalladamente cada uno de los pasos del algoritmo propuesto.

En el paso 2 se calcula la huella o ruido residual I_{ruido} de una imagen I eliminando el contenido de la escena de la imagen mediante un filtro de eliminación de ruido $\mathbf{F}$.

$$I_{ruido} = \quad I \quad - \quad F(I) \tag{4.1}$$

Entre los diferentes filtros que existen para la eliminación del ruido de las imágenes los que usan la transformada *wavelet* dan mejor resultado debido a que el ruido residual que se obtiene con este filtro contiene la menor cantidad de rasgos de la escena. Generalmente, las áreas alrededor de los bordes son mal interpretadas cuando se utilizan únicamente filtros de eliminación de ruido menos robustos, tales como el filtro de Wiener o el filtro de mediana. Por este motivo se seleccionó el filtro de eliminación de ruido basado en la transformada *wavelet*.

En el paso 1 del algoritmo para la eliminación del ruido se descompone la imagen en sus canales de color R, G y B. Para cada canal de color se realiza una descomposición *wavelet* en cuatro niveles de la imagen utilizando QMF Daubechies 8-tap. El número de niveles de la descomposición se puede variar modificando de esta manera el procesamiento requerido y también la calidad de la información extraída de las imágenes.

Una vez aplicada la descomposición *wavelet* de la imagen, para cada nivel de la descomposición se obtienen los componentes de alta frecuencia H, V y D. Se denota a los componentes de alta frecuencia como $c(i,j)$ donde $c \in \{H,V,D\}$ y donde (i,j) corre a través de un conjunto de índices J que depende del nivel de descomposición.

Para cada componente de alta frecuencia se calcula la varianza local usando la estimación *Maximum A-Posteriori Probability* (MAP). La información estadística de las imágenes puede diferir mucho entre ellas e incluso entre las diferentes regiones de una misma imagen, para combatir esta diferencia se utilizan estimadores adaptativos con diferentes

tamaños de ventana de vecindad y no sólo un tamaño de ventana.

En el algoritmo propuesto se plantea calcular la varianza local de forma adaptativa o no, dependiendo de las necesidades y/o restricciones computacionales.

En caso de requerir estimadores adaptativos para cada componente de alta frecuencia se calcula la varianza local para cuatro tamaños de ventana $W \times W$ de vecindario N con $W \in \{3, 5, 7, 9\}$. El parámetro σ_0, que controla qué tan fuerte será la supresión de ruido, toma el valor de 5 (ampliamente recomendado en la literatura).

$$\hat{\sigma}^2(i,j) = \text{máx}\left(0, \frac{1}{W^2}\sum_{(i,j)\in N} c^2(i,j) - \sigma_0^2\right), \quad (i,j) \in J \tag{4.2}$$

A continuación, se toma el mínimo de las 4 varianzas como la estimación final:

$$\hat{\sigma}^2(i,j) = \text{mín}\left(\sigma_3^2(i,j), \sigma_5^2(i,j), \sigma_7^2(i,j), \sigma_9^2(i,j)\right), \quad (i,j) \in J \tag{4.3}$$

En el caso de no requerir la utilización de estimadores adaptativos que requieren de más procesamiento se calcula la varianza local para un solo tamaño de ventana considerando $W = 3$.

Después, se calculan los componentes *wavelet* sin ruido utilizando el filtro de Wiener:

$$c_{limpio}(i,j) = \quad c(i,j)\frac{\hat{\sigma}^2(i,j)}{\hat{\sigma}^2(i,j) + \sigma_0^2} \tag{4.4}$$

Por último, la imagen final sin ruido se obtiene aplicado la transformada *wavelet* inversa a los componentes *wavelet* sin ruido de cada uno de los niveles.
Teniendo la imagen sin ruido se obtiene la huella del sensor.
La huella calculada I_{ruido} contiene todos los elementos que se presentan sistemáticamente en cada una de las imágenes, incluyendo algunos que no son causados por el sensor como las características causadas por la interpolación de colores o la compresión JPEG. La mayoría de estas características son generadas por el proceso de interpolación cromática dependiendo de la CFA utilizada por la cámara. Debido a que estas características tienen una naturaleza periódica se pueden eliminar mediante el paso 3.

En el paso 3 del algoritmo se limpia la huella de las características que no son intrínsecas al sensor aplicando un promediado a cero de filas y columnas como se sugiere en [CFGL08], de tal manera que los promedios de las filas y de las columnas sean iguales a cero. Esto se logra restando el promedio de la columna a cada píxel de la columna y posteriormente restando el promedio de la fila a cada píxel de la fila. Esta operación se aplica a todas las filas y columnas de la imagen.

En el paso 4 utilizando la ecuación 4.5 se le da un mayor peso al canal verde ya que debido a la configuración de la matriz de color este contiene más información sobre la imagen que el resto de los canales de color [CSA08, McK07, APS98].

$$I_{ruido} = \quad 0.3 \cdot I_{ruido_R} + 0.6 \cdot I_{ruido_G} + 0.1 \cdot I_{ruido_B} \tag{4.5}$$

Cuando se cuenta con N imágenes de la misma cámara es posible estimar el patrón del ruido del sensor, tal y como se hace en el paso 5 del algoritmo.

Recordando lo visto en la sección 2.1.3.2, la parte dominante del patrón del ruido es el ruido PRNU compuesto por el ruido PNU y los defectos de baja frecuencia. Dado que los componentes de baja frecuencia no son una característica propia del sensor, sólo se debe considerar el ruido PNU para la identificación del sensor ya que es una de sus características intrínsecas.

Una aproximación del ruido PNU se calcula mediante el promedio del ruido residual de varias imágenes:

$$P_{ruido} = \frac{1}{N}\sum_{i=1}^{N} I_{ruido} \tag{4.6}$$

Debido a la propiedad determinista del patrón de ruido del sensor que está presente en cada imagen capturada, se puede usar este patrón como huella para identificar el dispositivo que generó la imagen objeto en investigación [LFG06]. Haciendo una analogía, se puede decir que el patrón del ruido del sensor es para una cámara lo que la huella para un ser humano.

4.2.2 Algoritmo de Extracción de Características de la Imagen

Como se detalla en el Anexo B la máquina SVM requiere de una serie de características que representan los datos a clasificar. Es necesario extraer una serie de características de las huellas de imágenes de fuente conocida a priori para entrenar a la máquina SVM y de las imágenes de las que se requiere identificar la fuente. El algoritmo a seguir para extraer dichas características es el siguiente.

- Entrada: huella del sensor.
- Salida: 81 características representativas de la huella del sensor.

Paso 1. *Separar los canales de color la huella del sensor.*

Para cada canal de color:

Paso 2. *Hacer una descomposición wavelet de un nivel de la huella del sensor de la imagen.*

Para cada componente wavelet c_i donde $c \in \{H, V, D\}$:

Paso 3. *Calcular k momentos centrales con:*

$$m_k = \frac{1}{n}\sum_{i=1}^{n} |c_i - \overline{c}|^k$$

En este algoritmo los momentos centrales de los valores absolutos de cada componente de alto nivel de la huella del sensor son tomados como una medida de la distribución del ruido.

En el paso 1 se separa la huella del sensor en los 3 canales de color R, G y B. A continuación en el paso 2 se realiza una descomposición *wavelet* con 8-tap Daubechies para obtener los componentes de alta frecuencia H, V y D de los que se extraerán las características.

En el paso 3 para cada componente *wavelet* c_i se calcula el k-ésimo momento central absoluto utilizando la ecuación 4.7 para $k = 9$.

$$m_k = \frac{1}{n} \sum_{i=1}^{n} |c_i - \overline{c}|^k \tag{4.7}$$

Para cada componente *wavelet* H, V y D de cada canal de color se calculan 9 momentos resultando en un total de $3 \times 3 \times 9 = 81$ características como se ilustra en la Tabla 4.1.

Tabla 4.1: Momentos centrales característicos de la huella del sensor

Canal de color	Componente wavelet	Momentos centrales					
R	**H**	m_1	m_2	m_3	m_4	...	m_9
	V	m_1	m_2	m_3	m_4	...	m_9
	D	m_1	m_2	m_3	m_4	...	m_9
G	**H**	m_1	m_2	m_3	m_4	...	m_9
	V	m_1	m_2	m_3	m_4	...	m_9
	D	m_1	m_2	m_3	m_4	...	m_9
B	**H**	m_1	m_2	m_3	m_4	...	m_9
	V	m_1	m_2	m_3	m_4	...	m_9
	D	m_1	m_2	m_3	m_4	...	m_9

4.3 Algoritmo de Falsificación de la Identidad de una Imagen

Una vez que se tiene la posibilidad de eliminar el ruido del sensor y de extraer el patrón del ruido del sensor se puede plantear la falsificación de la identidad de una imagen. El algoritmo a seguir es el siguiente:

- Entradas:
  - Imagen víctima I.
  - N imágenes de superficies uniformemente iluminadas de la cámara suplantadora.
- Salida: Imagen falsificada I_{falsa}.

Paso 1. *Calcular I_{limpia} eliminando el ruido del sensor de la imagen víctima I.*
Paso 2. *Calcular el patrón del ruido del sensor P_{ruido} de N imágenes de la cámara suplantadora.*
Paso 3. *Realizar una descomposición wavelet de un nivel de I_{limpia} obteniendo los componentes L_I, H_I, V_I y D_I.*
Paso 4. *Realizar una descomposición wavelet de un nivel de P_{ruido} obteniendo los componentes H_P, V_P y D_P.*
Paso 5. *Calcular los componentes wavelet falsificados mediante:*

$$c_F = c_I + c_P \quad donde \quad c \in \{H, V, D\}$$

Paso 6. *Obtener I_{falsa} aplicando la transformada wavelet inversa con L_I, H_F, V_F y D_F.*

A continuación se describe detalladamente cada paso del algoritmo. En el paso 1 se calcula la Imagen I_{limpia} aplicando el paso 1 del algoritmo de la sección 4.2.1 a la imagen víctima I para eliminar los rastros de la cámara con la que fue adquirida.

A continuación, en el paso 2 con el algoritmo de la sección 4.2.1 se extrae el patrón P_{ruido} de N imágenes tomadas con la cámara con la que se desea llevar a cabo la suplantación.

Para tener una huella de mejor calidad y obtener mejores resultados en la falsificación se recomienda que el número N de imágenes sea superior a 50, y que las imágenes se hayan adquirido de superficies planas sin textura iluminadas uniformemente. Como superficies planas se pueden considerar fotografías del cielo despejado o de un papel blanco.

Después en el paso 3 se realiza una descomposición *wavelet* a la imagen víctima sin ruido I_{limpia} para obtener sus componentes *wavelet* de alta frecuencias L_I, H_I, V_I y D_I. Asimismo en el paso 4 se descompone la imagen P del patrón a implantar obteniendo los componentes *wavelet* H_P, V_P y D_P.

En el paso 5 se calculan los componentes *wavelet* de alta frecuencia que formarán la imagen falsificada sumando uno a uno los componentes *wavelet* H, V y D de la imagen I_{limpia} y de P_{ruido}.

Por último en el paso 6 se construye la imagen I_{falsa} con el componente de baja frecuencia de la imagen a atacar L_I y los componentes de alta frecuencia calculados en el paso 5.

Este proceso se ilustra en la Figura 4.1.

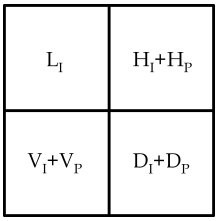


Figura 4.1: f_{falsa} Resultado de la suma de los componentes *wavelet*

El Anexo C contiene los detalles de implementación de estos algoritmos.

Capítulo 5

Experimentos y Resultados

En este capítulo se describen los experimentos realizados para evaluar la eficacia de los algoritmos propuestos y los resultados obtenidos.

5.1 Experimentos

5.1.1 Evaluación del Algoritmo de Identificación de la Fuente

Para evaluar la efectividad del algoritmo de identificación de la fuente de dispositivos móviles se realizaron los experimentos que a continuación se detallan.

5.1.1.1 Experimento 1

En este experimento se probó con un grupo de 3 cámaras digitales de dispositivos móviles cuya marca y modelo se muestran en la Tabla 5.1.

Tabla 5.1: Dispositivos utilizados en el experimento 1

Marca	Modelo
LG	E510f
	400
Samsung	GT-I8160P

De cada uno de los dispositivos se obtuvieron un conjunto de fotografías naturales, es decir, de cualquier tipo de escena sin ninguna restricción. Para este experimento usó la configuración por defecto del algoritmo de extracción de la huella. La tabla 5.2 resume los principales parámetros utilizados.

Tabla 5.2: Parámetros utilizados en el experimento 1

Parámetro	Valor
Tipo de Fotos	Naturales
Dimensiones	1024×1024
Número de Cámaras	3
Fotos entrenadas x Cámara	60, 60, 60, 70, 70, 70, 80, 80, 80, 90, 90, 90, 100, 100, 100
Fotos Probadas x Cámara	10, 20, 30, 10, 20, 30, 10, 20, 30, 10, 20, 30, 10, 20, 30
Promediado a Cero	Sí
Tamaño de Vecindario	3

Los resultados obtenidos en la identificación de la fuente se muestran en la Tabla 5.3, como se puede observar en la tabla el número de fotografías utilizadas para entrenar y probar afecta directamente el porcentaje de acierto. El número de fotografías para entrenar que arrojó mejores resultados fue 90 con las diferentes combinaciones de fotografías para probar. Al utilizar 100 fotografías para el entrenamiento se redujo la tasa de aciertos obteniendo prácticamente los mismos resultados que cuando se entrenó sólo con 60 fotografías. De lo anterior se puede concluir que el hecho de entrenar con el mayor número de fotografías posibles no siempre implicará mejores resultados. En cuanto al número de fotografías para entrenar se puede concluir que a pesar de que con 10 fotografías se obtienen los mejores resultados, éste es un número muy pequeño como para representar escenarios reales completamente.

Tabla 5.3: Matriz de confusión con resultados del experimento 1

		Fotos entrenadas				
		60	**70**	**80**	**90**	**100**
Tasa de aciertos		91,39 %	90,24 %	90,21 %	91,36 %	91,39 %
Fotos	10	88,33 %	88,33 %	90 %	90 %	88,33 %
Probadas	20	87,5 %	81,67 %	82,5 %	88,33 %	87,5 %
	30	82,78 %	78,89 %	80,56 %	84,44 %	82,78 %

5.1.1.2 Experimento 2

En este experimento se probó con un grupo de 8 cámaras digitales de dispositivos móviles de 4 diferentes fabricantes cuya marca y modelo se muestran en la Tabla 5.4.

Tabla 5.4: Dispositivos utilizados en el experimento 2

Marca	Modelo
Apple	iPhone3G
	iPhone4S
	iPad2
Blackberry	8520
Sony Ericsson	UST25a
	U5I
Samsung	GTI9100
	GTS5830

En la Tabla 5.5 se resumen los parámetros principales utilizados en este experimento.

Tabla 5.5: Parámetros utilizados en el experimento 2

Parámetro	Valor
Tipo de Fotos	Naturales
Dimensiones	1024×1024
Número de Cámaras	8
Fotos entrenadas x Cámara	100
Fotos Probadas x Cámara	100
Promediado a Cero	Sí
Tamaño de Vecindario	3

En este experimento los algoritmos propuestos obtuvieron un total de porcentaje de acierto del **93,625 %** al identificar entre marca y modelo. Los resultados de este experimento se muestran en la Tabla 5.6 donde se puede apreciar que es posible la identificación entre diferentes modelos de la misma marca.

Tabla 5.6: Matriz de confusión con resultados del experimento 2

		Apple			**Blackberry**	**Sony Ericsson**		**Samsung**	
		iPhone3G	**iPhone4S**	**iPad2**	**8520**	**UST25a**	**U5I**	**GTI9100**	**GTS5830**
Apple	**iPhone3G**	**92 %**	1 %	0 %	0 %	0 %	1 %	0 %	6 %
	iPhone4S	0 %	**96 %**	0 %	0 %	1 %	0 %	3 %	0 %
	iPad2	0 %	0 %	**99 %**	0 %	0 %	0 %	1 %	0 %
Blackberry	**8520**	0 %	0 %	0 %	**94 %**	0 %	2 %	0 %	4 %
Sony Ericsson	**UST25a**	7 %	2 %	0 %	0 %	**91 %**	0 %	0 %	0 %
	U5I	2 %	0 %	0 %	1 %	0 %	**94 %**	1 %	2 %
Samsung	**GTI9100**	4 %	8 %	0 %	0 %	0 %	5 %	**83 %**	0 %
	GTS5830	0 %	0 %	0 %	0 %	0 %	0 %	0 %	**100 %**

5.1.1.3 Experimento 3

En este experimento se probó con un grupo de 14 cámaras digitales de dispositivos móviles de 7 diferentes fabricantes cuya marca y modelo se muestran en la Tabla 5.7.

Tabla 5.7: Dispositivos utilizados en el experimento 3

Marca	Modelo	Alias
Apple	iPhone3G	A1
	iPhone4S	A2
	iPad2	A3
	IpodTouch	A4
Blackberry	8520	B1
Sony Ericsson	UST25a	SE1
	U5I	SE2
Samsung	GTI9100	S1
	GTS5830	S2
	GT-S5830M	S3
Lg	E400	L
HTC	DesireHD	H1
	Desire	H2
Nokia	E61I	N

En la Tabla 5.8 se resumen los parámetros principales utilizados en este experimento.

Tabla 5.8: Parámetros utilizados en el experimento 3

Parámetro	Valor
Tipo de Fotos	Naturales
Dimensiones	1024×1024
Número de Cámaras	14
Fotos entrenadas x Cámara	100
Fotos Probadas x Cámara	100
Promediado a Cero	Sí
Tamaño de Vecindario	3

En este experimento los algoritmos propuestos obtuvieron un total de porcentaje de acierto del **87,214 %** al identificar entre marca y modelo. Los resultados de este experimento se muestran en la Tabla 5.9.

Tabla 5.9: Matriz de confusión con resultados del experimento 3

		Apple				**Blackberry**	**Sony Ericsson**		**Samsung**			**Lg**	**HTC**		**Nokia**
		A1	**A2**	**A3**	**A4**	**B**	**SE1**	**SE2**	**S1**	**S2**	**S3**	**L**	**H1**	**H2**	**N**
Apple	**A1**	**90 %**	0 %	0 %	2 %	0 %	0 %	0 %	0 %	7 %	0 %	1 %	0 %	0 %	0 %
	A2	0 %	**91 %**	0 %	3 %	0 %	0 %	0 %	3 %	0 %	0 %	0 %	1 %	2 %	0 %
	A3	0 %	0 %	**98 %**	0 %	0 %	0 %	0 %	2 %	0 %	0 %	0 %	0 %	0 %	0 %
	A4	0 %	0 %	1 %	**88 %**	0 %	0 %	0 %	0 %	0 %	0 %	3 %	6 %	0 %	2 %
Blackberry	**B**	0 %	0 %	0 %	2 %	**73 %**	0 %	0 %	0 %	4 %	0 %	0 %	1 %	0 %	20 %
Sony	**SE1**	7 %	0 %	0 %	0 %	0 %	**80 %**	0 %	0 %	0 %	0 %	1 %	12 %	0 %	0 %
Ericsson	**SE2**	1 %	0 %	0 %	2 %	2 %	0 %	**86 %**	1 %	2 %	5 %	1 %	0 %	0 %	0 %
Samsung	**S1**	4 %	5 %	0 %	4 %	0 %	0 %	1 %	**83 %**	0 %	0 %	1 %	0 %	2 %	0 %
	S2	0 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	**100 %**	0 %	0 %	0 %	0 %	0 %
	S3	0 %	0 %	1 %	0 %	0 %	0 %	8 %	0 %	0 %	**85 %**	0 %	1 %	0 %	5 %
Lg	**L**	0 %	0 %	0 %	9 %	0 %	6 %	0 %	0 %	2 %	0 %	**70 %**	13 %	0 %	0 %
HTC	**H1**	2 %	0 %	0 %	0 %	0 %	11 %	0 %	0 %	1 %	0 %	1 %	**85 %**	0 %	0 %
	H2	0 %	6 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	**94 %**	0 %
Nokia	**N**	0 %	0 %	0 %	0 %	2 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	**98 %**

5.1.2 Evaluación de Algoritmos de Falsificación de la Identidad

En esta sección describen los experimentos realizados con los algoritmos de eliminación de la huella del sensor (paso 1 del algoritmo 4.2.1) y de falsificación de la huella de la cámara fuente (algoritmo 4.3).

En estos experimentos se realizó la eliminación y la falsificación de las huellas con las implementaciones propuestas y con la herramienta "PRNU Decompare" [DEC]. Esta herramienta permite la eliminación y la suplantación del patrón del sensor [DEC], para llevar a cabo la eliminación y la suplantación de la huella este programa requiere como entrada una fotografía de un marco oscuro y un número N de imágenes de superficies planas iluminadas uniformemente (se recomienda un mínimo de 30 imágenes).

Los resultados obtenidos se compararon haciendo uso la herramienta "NFI PRNU Compare" [COMa], esta herramienta permite comparar los patrones del ruido del sensor de varias imágenes.

A continuación se detallan cada uno de los experimentos con sus resultados.

5.1.2.1 Experimento de Eliminación de la Identidad de una Imagen

Para este experimento se utilizaron las fotografías de 3 cámaras digitales de dispositivos móviles cuya marca y modelo se muestran en la Tabla 5.10.

Tabla 5.10: Dispositivos utilizados para la eliminación de la identidad

Marca	Modelo
LG	E510f
	400
Samsung	GT-I8160P

De cada uno de los dispositivos se obtuvieron 50 fotografía de imágenes planas uniformemente iluminadas, 1 fotografía totalmente oscura cubriendo totalmente el lente de la cámara y 1 fotografía seleccionada al azar de la base de datos de fotografías. Todas las fotografías fueron recortadas a un tamaño de 1024 x 1024.

A continuación se procedió a generar el primer grupo de imágenes sin huella, haciendo uso de la implementación de eliminación del ruido del sensor propuesto con los parámetros de configuración por defecto. Cabe remarcar que para realizar esta eliminación no se requirió de fotografías adicionales a la fotografía de la que se pretendía eliminar el ruido del sensor.

Después se generó el segundo grupo de imágenes sin huella con la herramienta "PRNU Decompare", dando como entrada a este programa las 50 imágenes planas y la del marco oscuro.

Para evaluar la efectividad del algoritmo de eliminación y falsificación de la huella se compararon los dos grupos de imágenes con la herramienta "NFI PRNU Compare", los

resultados se muestran en la Tabla 5.11.

Tabla 5.11: Comparativa entre patrones e imágenes sin ruido

Patrón LG-E510f	**Rojo**	**Verde**	**Azul**	**Suma**
Patrón LG-E510f	1	1	1	3
Foto original	-0,014645672	-0,0017777978	-0,007864626	-0,024288096
Foto sin ruido - Propuesta	-0,015506644	-0,003044259	-0,008411303	-0,026962206
Foto sin ruido - Decompare	-0,018929206	-0,0023383496	-0,012027217	-0,033294775
Patrón Samsung-GTI8160P	**Rojo**	**Verde**	**Azul**	**Suma**
Patrón Samsung GT-I8160P	1	1	1	3
Foto sin ruido - Propuesta	0,0051651257	0,005551344	0,0042196396	0,01493611
Foto original	0,004623602	0,0050348267	0,0030041975	0,012662627
Foto sin ruido - Decompare	-0,0012833464	-0,001952231	-0,0026684676	-0,005904045
Patrón LG-E400	**Rojo**	**Verde**	**Azul**	**Suma**
Patrón LG-E400	1	1	1	3
Foto original	0,011481647	0,010190065	0,01825918	0,039930895
Foto sin ruido - Propuesta	0,010315191	0,008225861	0,017940063	0,036481116
Foto sin ruido - Decompare	0,010638827	0,009045472	0,016430777	0,036115076

En la Tabla 5.11 se muestran los resultados de comparar cada patrón del ruido del sensor de cada una de las cámaras con las imágenes sin ruido generadas por las dos herramientas y contra la fotografía original. La herramienta "PRNU Decompare" nos permite hacer comparaciones midiendo que tanto se parece un patrón a otro, las filas que están más cerca del patrón con el que se compara son las que más se le asemejan.

En las 3 pruebas las imágenes sin ruidos generadas con "Decompare" resultaron ser las menos parecidas al patrón, esto era de esperarse ya que consideran mayor número de información para eliminar la huella.

De manera sorprendente para la cámara Samsung-GTI8160P la fotografía sin ruido generada por la propuesta de este trabajo resultó ser más parecida al patrón que la misma fotografía original. Posiblemente en el caso de esta fotografía el contenido de la fotografía tenga alguna influencia.

En el caso de la cámara LG-400, los resultados de las comparaciones de las imágenes sin ruido tuvieron resultados muy similares, lo que indica que en este caso el algoritmo propuesto obtuvo buenos resultados acercándose al resultado del programa "Decompare" pero sin la necesidad de usar la fotografía del marco oscuro, ni las 30 imágenes planas.

5.1.2.2 Experimento de Falsificación de la Identidad de una Imagen

Para este experimento utilizaron el mismo conjunto de fotografías que en el experimento de la eliminación del ruido en la sección 5.1.2.1. De forma similar al experimento anterior se extrajo la huella de una de las cámaras y se inyectó a las otras dos haciendo

uso del algoritmo propuesto y de la herramienta “Decompare”, después se compararon los resultados con la herramienta “NFI PRNU Compare”.

Los roles que jugaron cada una de las cámaras se muestran en la Tabla 5.12.

Tabla 5.12: Dispositivos utilizados para la falsificación de la identidad

Marca	Modelo	Rol
LG	E510f	Suplantadora
Samsung	GT-I8160P	Víctima 2
LG	400	Víctima 1

Para realizar la falsificación del patrón del ruido del sensor con el algoritmo propuesto en este trabajo únicamente se requirieron las 50 imágenes planas uniformemente ilumina-das pertenecientes a la cámara suplantadora.

En caso de la herramienta “Decompare” para realizar la falsificación el programa requiere como entrada las 50 fotografías planas y la fotografía totalmente oscura tanto de la cámara suplantadora como de la cámara víctima. Después de realizar la falsificación en las dos cámaras víctimas se compararon los resultados que están resumidos en la Tabla 5.13.

Tabla 5.13: Comparativa entre patrones, imágenes originales y víctimas

Comparación con Patrón Suplantadora	Rojo	Verde	Azul	Suma
Patrón suplantadora	1	1	1	3
Víctima1 Falsificada -Decompare	0,009219962	0,0054620425	0,009098741	0,023780745
Víctima1 Falsificada -Propuesta	0,007900867	0,0046529872	0,0083521	0,020905953
Víctima1 Original	0,0073570074	0,004183661	0,0075896666	0,019130334
Comparación con Patrón Suplantadora	**Rojo**	**Verde**	**Azul**	**Suma**
Patrón suplantadora	1	1	1	3
Víctima2 Falsificada -Decompare	0,01418404	0,013986045	0,013574668	0,041744754
Víctima2 Original	0,011300047	0,013949845	0,0125216115	0,0377715
Víctima2 Falsificada -Propuesta	0,008964902	0,0066337977	0,004440412	0,020039111

En el caso de la víctima 1 en la Tabla 5.13 se puede observar que las dos suplantaciones resultaron tener mayor similitud con el patrón de la cámara suplantadora y el resultado de “Decompare” se acerca más aunque la diferencia no es muy significativa considerando que ellos utilizan un número mucho mayor de imágenes como fuente de información.

En el caso de la víctima 2 el resultado del algoritmo propuesto fue el que menos similitud tuvo con el patrón de la cámara suplantadora.

Los resultados obtenidos hasta el momento eran esperados, debido a que el algoritmo propuesto en este trabajo no sume que se tiene acceso a la cámara fuente y en el trabajo

de “Decompare” sí. Es importante notar que en escenarios reales normalmente no se tiene acceso a la cámara víctima.

Capítulo 6

Conclusiones y Trabajo Futuro

6.1 Conclusiones

En el presente trabajo se han estudiado las diferentes técnicas del análisis forense de imágenes digitales con especial atención en las orientadas a la identificación de la fuente de una imagen. Luego de realizar un análisis comparativo de los trabajos realizados para la identificación de la fuente en los últimos años, se detectó la necesidad de enfocar éstas a las técnicas de identificación de fuente en cámaras digitales de dispositivos móviles, debido a que el número de este tipo de dispositivos está en constante crecimiento y prácticamente han remplazado a las cámaras digitales tradicionales. Para diseñar los algoritmos enfocados a las cámaras integradas en dispositivos móviles se estudiaron las diferencias entre este tipo de cámaras y las cámaras tradicionales, analizando los componentes que permiten obtener las características que identifican de manera única a una cámara digital. Se encontró que de acuerdo a la estructura y funcionamiento de las cámaras digitales de dispositivos móviles las técnicas más adecuadas para realizar análisis forense en ellas son las que se basan en el ruido del sensor y las que utilizan las transformadas *wavelet*. Estas técnicas utilizan como medio de identificación la información que está presente con mayor fuerza en los dispositivos móviles.

En virtud de lo anteriormente expuesto se propuso un algoritmo para la identificación de los dispositivos móviles fuente combinando las técnicas basadas en la huella del sensor y en la transformación *wavelet*.

Este algoritmo se compone, a su vez, de diversos algoritmos.

Así, primero se propuso un algoritmo que permite obtener la huella de una imagen de manera individual a través de eliminación de ruido generado por el sensor de la cámara fuente en la imagen.

A continuación se diseñó un segundo algoritmo que permite extraer el patrón de la huella de una serie imágenes pertenecientes a la cámara del dispositivo móvil fuente.

Después se propuso un algoritmo basado en la transformación *wavelet* para extraer las características tanto de la huella como del patrón de la huella y así poder clasificarlas mediante el uso de las máquinas SVM.

Al estudiar las diferentes técnicas de análisis forense se encontró que al diseñarlas en su mayoría no consideran los posibles ataques que personas mal intencionadas podrían realizar para evitar que logren su objetivo o para falsificar sus resultados. Por este motivo se realizó un estudio de las diferentes técnicas de ataques a las técnicas de identificación de la fuente así como la forma de identificarlos. De este análisis y del diseño de los algoritmos propuestos para la identificación de la fuente en dispositivos se derivó el desarrollo de un algoritmo más que permite falsificar la identidad una imagen inyectando el patrón de huellas o la huella de una imagen en una imagen víctima.

La característica más importante de los algoritmos propuestos es que no requieren tener acceso a la cámara fuente con la que se generó una imagen para poder identificarla o para poder falsificarla de una manera efectiva. Una aplicación de utilidad al algoritmo de la eliminación de la huella del sensor es la posibilidad de ocultar la identidad de las fotografías de las personas, de esta manera antes de publicar sus fotografías en las redes sociales podrían ejecutar este algoritmo y así evitar ser víctimas al robo de identidad y la posibilidad de ser culpados injustamente.

6.2 Trabajo Futuro

Como posibles trabajos futuros pueden señalarse los siguientes:

- Realizar experimentos de identificación de la fuente utilizando diferentes regiones de interés de la imagen. En los experimentos realizados se utilizaron los 1024 x 1024 píxeles centrales de las imágenes. Existe la posibilidad de obtener mejores resultados considerando diferentes áreas de la imagen como el área central más las áreas de las esquinas o únicamente el área de alguna de las esquinas en lugar de la central.

- Robustecer los algoritmos de identificación de la fuente diseñados contra posibles ataques. Como se probó en los experimentos de eliminación y falsificación de la huella es posible inhabilitar o engañar a las técnicas de identificación de la fuente, por lo que un área a desarrollar es el diseño de estrategias para detectar cuando una huella ha sido eliminada o suplantada.

- Extender la clasificación de la fuente haciendo uso de técnicas de correlación basadas en métricas *Peak to Correlation Energy* (PCE) que al ser una medida de similitud entre dos señales permitirían realizar la identificación si la necesidad de un entrenamiento con las clases conocidas a priori.

- Extender el algoritmo de identificación de la fuente a técnicas de agrupación en las que dado un conjunto de imágenes se agrupan y generan clases de acuerdo a la fuente de la que provienen, siendo ésta una posibilidad más para evitar el entrenamiento previo.

6.3 Publicaciones

Además de los resultados obtenidos de los experimentos, la elaboración de este trabajo dio lugar a dos publicaciones. Del estudio del estado del arte se originó la publicación:

- **"Techniques for Source Camera Identification"** (con Ana Lucila Sandoval Orozco, David Manuel Arenas González, Luis Javier García Villalba, Julio César Hernández Castro), in proceedings of the ***6th International Conference on Information Technology***. Amman, Jordan, May 8-10, 2013 [SOAGRC+13b].

Y de la propuesta del algoritmo para la identificación de la fuente se derivó la publicación:

- **"Source Identification for Mobile Devices, based on Wavelet Transforms Combined with Sensor Imperfections"** (con Ana Lucila Sandoval Orozco, David Manuel Arenas González, Luis Javier García Villalba, Julio César Hernández Castro), ***Computing***, 2013:1-11, February, 2013 [SOAGRC+13a].

Bibliografía

[AKMS05] I. Avcibas, M. Kharrazi, N. Memon, and B. Sankur. Image Steganalysis with Binary Similarity Measures. *EURASIP Journal on Advances in Signal Processing*, 2005(17):2749–2757, Juanuary 2005.

[AM12] T. Ahonen and A. Moore. Tomi Ahonen Almanac 2012: Mobile Telecoms Industry Annual Review, 2012.

[AMS03] I. Avcibas, N. Memon, and B. Sankur. Steganalysis Using Image Quality Metrics. *IEEE Transactions on Image Processing*, 12(2):221–229, February 2003.

[AP13] J. Adams and B Pillman. Digital camera image formation: Introduction and hardware. In Husrev Taha Sencar and Nasir Memon, editors, *Digital Image Forensics*, pages 3–44. Springer, August 2013.

[APS98] J. Adams, K. Parulski, and K. Spaulding. Color Processing in Digital Cameras. *Micro, IEEE*, 18(6):20–30, December 1998.

[Are11] D. Arenas. Análisis Forense de Imágenes de Móviles Mediante el Uso de Metadatos. Masters Thesis 13507, Universidad Complutense de Madrid, November 2011.

[Ass] Adobe Developers Association. TIFF Revision 6.0, June 3, 1992, http://partners.adobe.com/public/developer/en/tiff/TIFF6.pdf.

[AZ06] M. Al-Zarouni. Mobile Handset Forensic Evidence: a Challenge for Law Enforcement. In *Proceedings of the 4th Australian Digital Forensics Conference*. School of Computer and Information Science, Edith Cowan University, December 2006.

[Bae10] R. Baer. Resolution Limits in Digital Photography: The Looming End of the Pixel Wars - OSA technical Digest (CD). In *Proceedings of the Imaging Systems*, page ITuB3. Optical Society of America, June 2010.

[BL04] M. Boutell and J. Luo. Photo Classification by Integrating Image Content and Camera Metadata. In *Proceedings of the 17th International Conference on Pattern Recognition*, volume 4, pages 901–904. IEEE Computer Society, August 2004.

[BL05] M. Boutell and J. Luo. Beyond Pixels: Exploiting Camera Metadata for Photo Classification. *Pattern Recognition*, 38(6):935–946, June 2005.

[BSM06] S. Bayram, H. T. Sencar, and N. Memon. Improvements on Source Camera-Model Identification Based on CFA Interpolation. In *Working Group 11.9 International Conference on Digital Forensics*, pages 24–27. Springer, February 2006.

[BSM08] S. Bayram, H. T. Sencar, and N. Memon. Classification of Digital Camera-Models Based on Demosaicing Artifacts. *Digital Investigation*, 5(1-2):49–59, September 2008.

[CAS+06] O. Celiktutan, I. Avcibas, B. Sankur, N. P. Ayerden, and C. Capar. Source Cell-Phone Identification. In *Proceedings of the IEEE 14th Signal Processing and Communications Applications*, pages 1–3. IEEE, April 2006.

[CCD] Sensores con Tecnología CCD vs CMOS, http://www.xatakafoto.com/camaras/sensores-con-tecnologia-ccd-vs-cmos.

[CD99] E. J. Candes and D. L. Donoho. Ridgelets: A Key to Higher-Dimensional Intermittency? *Philosophical Transactions of the London Royal Society*, 357:2495–2509, September 1999.

[CESR12] F. D. O. Costa, M. Eckmann, W. J. Scheirer, and A. Rocha. Open Set Source Camera Attribution. In *Proceedings of the 25th Conference on Graphics, Patterns and Images*, pages 71–78. IEEE, August 2012.

[CFGL08] M. Chen, J. Fridrich, M. Goljan, and J. Lukas. Determining Image Origin and Integrity Using Sensor Noise. *IEEE Transactions on Information Forensics and Security*, 3(1):74–90, March 2008.

[Cho06] K. S. Choi. Source Camera Identification Using Footprints From Lens Aberration. In *Proceedings on Digital Photography II*, number 852 in 6069, pages 60690J–60690J–8. SPIE International Society For Optical Engineering, February 2006.

[CK09] H. Cao and A. C. Kot. Accurate Detection of Demosaicing Regularity for Digital Image Forensics. *Transactions on Information Forensics and Security*, 4(4):899–910, December 2009.

[CL] C. C. Chang and C. J. Lin. LIBSVM: A Library for Support Vector Machines. Version 3.17, Abril 26, 2013, http://www.csie.ntu.edu.tw/cjlin/libsvm/.

[CLW06] K. S. Choi, E. Y. Lam, and K. Y. Wong. Automatic Source Camera Identification Using the Intrinsic Lens Radial Distortion. *Optics Express*, 14(24):11551–11565, November 2006.

[COMa] NFI PRNU Compare, http://sourceforge.net/projects/prnucompare/.

[Comb] Standarization Committee. Exchangeable Image File for digital still cameras: Exif version 2.3, April 26, 2010, http://www.cipa.jp/english/hyoujunka/kikaku/pdf/DC-008-2010_E.pdf.

[CSA08] O. Celiktutan, B. Sankur, and I. Avcibas. Blind Identification of Source Cell-Phone Model. *IEEE Transactions on Information Forensics and Security*, 3(3):553–566, September 2008.

[CSS07] W. Chen, Y.Q. Shi, and W. Su. Image Splicing Detection Using 2-D Phase Congruency and Statistical Moments of Characteristic Function. In *Proceedings on Security, Steganography, and Watermarking of Multimedia Contents IX*, volume 6505, pages 65050R–65050R–8. SPIE-International Society for Optical Engine, January 2007.

[DEC] PRNU Decompare, http://prnudecompare.sourceforge.net/.

[DHS01] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* Wiley, New York, USA, 2nd edition, 2001.

[DSM07] A. E. Dirik, H. T. Sencar, and N. Memon. Source Camera Identification Based on Sensor Dust Characteristics. In *Workshop on Signal Processing Applications for Public Security and Forensics*, pages 1–6. IEEE, April 2007.

[DV05] M.N. Do and M. Vetterli. The Contourlet Transform: an Efficient Directional Multiresolution Image Representation. *IEEE Transactions on Image Processing*, 14(12):2091–2106, December 2005.

[GBK+01] Z. J. Geradts, J. Bijhold, M. Kieft, K. Kurosawa, K. Kuroki, and N. Saitoh. Methods for Identification of Images Acquired with Digital Cameras. In *Proceedings on Enabling Technologies for Law Enforcement and Security*, volume 4232, pages 505–512. SPIE-International Society for Optical Engine, February 2001.

[GFC11] M. Goljan, J. Fridrich, and Mo Chen. Defending Against Fingerprint-Copy Attack in Sensor-Based Camera Identification. *IEEE Transactions on Information Forensics and Security*, 6(1):227–236, March 2011.

[GKWB07] T. Gloe, M. Kirchner, A. Winkler, and R. Bohme. Can We Trust Digital Image Forensics? In *Proceedings of the 15th International Conference on Multimedia*, pages 78–86. ACM Press, September 2007.

[Ham] E. Hamilton. JPEG File Interchange Format. Version 1.02, September 1, 1992, http://www.w3.org/Graphics/JPEG/jfif3.pdf.

[HCL03] C. W. Hsuand, C. C. Chang, and C. J. Lin. A Practical Guide to Support Vector Classification. Practical guide, Department of Computer Science and Information Engineering, National Taiwan University, April 2003.

[HKT07] R. Hain, C. J. Kahler, and C. Tropea. Comparison of CCD, CMOS and intensified cameras. *Experiments in Fluids*, 42(3):403–411, Juanuary 2007.

[HL07] G. C. Holst and T. S. Lomheim. *CMOS/CCD Sensors and Camera Systems*, volume 172 of *SPIE PM.* JCD Publishing, October 2007.

[IPT] International Press Telecomunications Council, http://www.iptc.org.

[KMC+06] N. Khannaa, A. K. Mikkilinenib, G. T. Chiub, J.P. Allebacha, and E. J. Delpa. Forensic Classification of Imaging Sensor Types. Rfc, Purdue University, February 2006.

[LC09] C. L. Lai and Y. S. Chen. The Application of Intelligent System to Digital Image Forensics. In *International Conference on Machine Learning and Cybernetics*, volume 5, pages 2991–2998. IEEE, July 2009.

[LF03] J. Lukas and J. Fridrich. Estimation of Primary Quantization Matrix in Double Compressed JPEG Images. In *Digital Forensic Research Workshop*, pages 5–8, August 2003.

[LF06] S. Lyu and H. Farid. Steganalysis Using Higher-Order Image Statistics. *IEEE Transactions on Information Forensics and Security*, 1(1):111–119, March 2006.

[LFG06] J. Lukas, J. Fridrich, and M. Goljan. Digital Camera Identification from Sensor Pattern Noise. *IEEE Transactions on Information Forensics and Security*, 1(2):205–214, June 2006.

[LH06] Y. Long and Y. Huang. Image Based Source Camera Identification using Demosaicking. In *Proceedings of the IEEE 8th Workshop on Multimedia Signal Processing*, pages 419–424. IEEE, October 2006.

[LLC+12] Q. Liu, X. Li, L. Chen, H. Cho, A. P. Cooper, Z. Chen, M. Qiao, and A. H. Sung. Identification of Smartphone-Image Source and Manipulation. In He Jiang, Wei Ding, Moonis Ali, and Xindong Wu, editors, *Advanced Research in Applied Artificial Intelligence*, volume 7345 of *Lecture Notes in Computer Science*, pages 262–271. Springer Berlin Heidelberg, Dalian, China, June 2012.

[LP05] R. Lukac and K. N. Plataniotis. Color Filter Arrays: Design and Performance Analysis. *IEEE Transactions on Consumer Electronics*, 51(4):1260–1267, November 2005.

[McK07] C. McKay. Forensic Analysis of Digital Imaging Devices. Technical report, University of Maryland, 2007.

[MKY08] F. J. Meng, X. W. Kong, and X. G. You. Source Camera Identification Based on Image Bi-Coherence and Wavelet Features. In *Proceedings of the Fourth Annual IFIP WG 11.9 International Conference on Digital Forensics*, Kyoto, Japan, January 2008.

[MSGW08] C. Mckay, A. Swaminathan, H. Gou, and M. Wu. Image Acquisition Forensics: Forensic Analysis to Identify Imaging Source. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, International Conference on Acoustics Speech and Signal Processing (ICASSP), pages 1657–1660. IEEE, June 2008.

[MSM04] K. L. Mehdi, H. T. Sencar, and N. Memon. Blind Source Camera Identification. In *Proceedings of the International Conference on Image Processing*, volume 1, pages 709–712. IEEE, October 2004.

[Nak05] J. Nakamura. *Image Sensors and Signal Processing for Digital Still Cameras*. CRC Press, Boca Raton, FL, USA, August 2005.

[OA11] L. Ozparlak and I. Avcibas. Differentiating Between Images Using Wavelet-Based Transforms: A Comparative Study. *IEEE Transactions on Information Forensics and Security*, 6(4):1418–1431, December 2011.

[PF05] A.C. Popescu and H. Farid. Exposing Digital Forgeries by Detecting Traces of Resampling. *IEEE Transactions on Signal Processing*, 53(2):758–767, February 2005.

[Pla00] J. Platt. AutoAlbum: Clustering Digital Photographs Using Probabilistic Model Merging. In *Proceedings of the IEEE Workshop on Content-based Access of Image and Video Libraries*, pages 96–100. IEEE, June 2000.

[PNK94] P. Pudil, J. Novovičová, and J. Kittler. Floating Search Methods in Feature Selection. *Pattern Recognition Letters*, 15(11):1119–1125, November 1994.

[RCC+08] N. L. Romero, V. G. Chornet, J. S. Cobos, A. S. Carot, F. C. Centellas, and M. C. Mendez. Recovery of Descriptive Information in Images From Digital Libraries by Means of EXIF Metadata. *Library Hi Tech*, 26(2):302–315, 2008.

[RH99] T. Randen and J. H. Husøy. Filtering for Texture Classification: A Comparative Study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(4):291–310, April 1999.

[RSYD05] R. Ramanath, W. E. Snyder, Y. Yoo, and M. S. Drew. Color Image Processing Pipeline. *Signal Processing Magazine, IEEE*, 22(1):34–43, January 2005.

[Sch12] H. G. Schaathu. *Machine Learning in Image Steganalysis*. Jhon Wiley and Sons, New York, USA, 2012.

[SLFK10] M. Steinebach, H. Liu, P. Fan, and S. Katzenbeisser. Cell Phone Camera Ballistics: Attacks and Countermeasures. In *Proceedings on Multimedia on Mobile Devices*, pages 75420B–75420B–9. SPIE-International Society for Optical Engine, January 2010.

[SOAGRC+13a] A.L. Sandoval Orozco, D.M. Arenas González, J. Rosales Corripio, L.J. García Villalba, and J.C. Hernandez-Castro. Source Identification for Mobile Devices, based on Wavelet Transforms Combined with Sensor Imperfections. *Computing*, pages 1–13, February 2013.

[SOAGRC+13b] A.L. Sandoval Orozco, D.M. Arenas González, J. Rosales Corripio, L.J. García Villalba, and J.C. Hernandez-Castro. Techniques for Source Camera Identification. In *Proceedings of the 6th International Conference on Information Technology*, pages 1–9, May 2013.

[Tes05] J. Tesic. Metadata Practices for Consumer Photos. *IEEE Multimedia*, 12(3):86–92, September 2005.

[TLL07] M. J. Tsai, C. L. Lai, and J. Liu. Camera/Mobile Phone Source Identification for Digital Forensics. In *Proceedings of the International Conference on Acoustics Speech and Signal Processing*, pages II–221–II–224. IEEE, April 2007.

[TNC10] V. L. L. Thing, K. Y. Ng, and E. C. Chang. Live Memory Forensics of Mobile Phones. *Digital Investigation*, 7:S74–S82, August 2010.

[VCEK07] T. Van Lanh, K. S. Chong, S. Emmanuel, and M. S. Kankanhalli. A Survey on Digital Camera Image Forensic Methods. In *Proceedings of the IEEE International Conference on Multimedia and Expo*, pages 16–19. IEEE, July 2007.

[WGKM09] B. Wang, Y. Guo, X. Kong, and F. Meng. Source Camera Identification Forensics Based on Wavelet Features. In *Proceedings of the International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, volume 0, pages 702–705. IEEE Computer Society, September 2009.

[XMP] XMP Specification: Adobe XML Metadata Framework, http://www.adobe.com/products/xmp.

Anexos

Anexo A

Transformada Wavelet en Imágenes Digitales

En esta sección se da una introducción a las transformadas *wavelet* y su uso en el área del tratado de imágenes digitales, ya que éstas juegan un papel importante en los algoritmos propuestos.

La descomposición *wavelet* permite separar señales en diferentes frecuencias, de esta manera se puede analizar y/o modificar sólo la información de las frecuencias que son de algún interés en particular. Una de las características más interesantes y útiles de las transformadas *wavelet* es que son reversibles, de esta manera una vez que se modifican las frecuencias deseadas se puede reconstruir la señal.

En el área de imágenes digitales la información de la imagen se descompone en diferentes frecuencias que contienen distintos niveles de detalle sobre la imagen. En particular para la eliminación del ruido, éste se aísla del resto de la imagen para poder atacarlo directamente sin alterar los detalles significativos de la imagen.

Como se puede observar en la Figura A.1(b) el componente *wavelet* de la esquina superior izquierda es una reproducción exacta de la imagen original A.1(a) en una menor resolución, a este componente se le llama componente de baja frecuencia L. El resto de los componentes son llamados componentes de alta frecuencia y como su nombre lo indica contienen la información de las frecuencias altas. Los componentes de alta frecuencia llamados H, V, y D representan los cambios locales en diferentes direcciones horizontal, vertical y diagonal respectivamente. Para generar los diferentes niveles de la descomposición *wavelet* se aplica la función recursivamente como se puede observar en la Figura A.1(d).

En la Figura A.1(c) se puede observar la descomposición en 4 niveles de la imagen A.1(a).

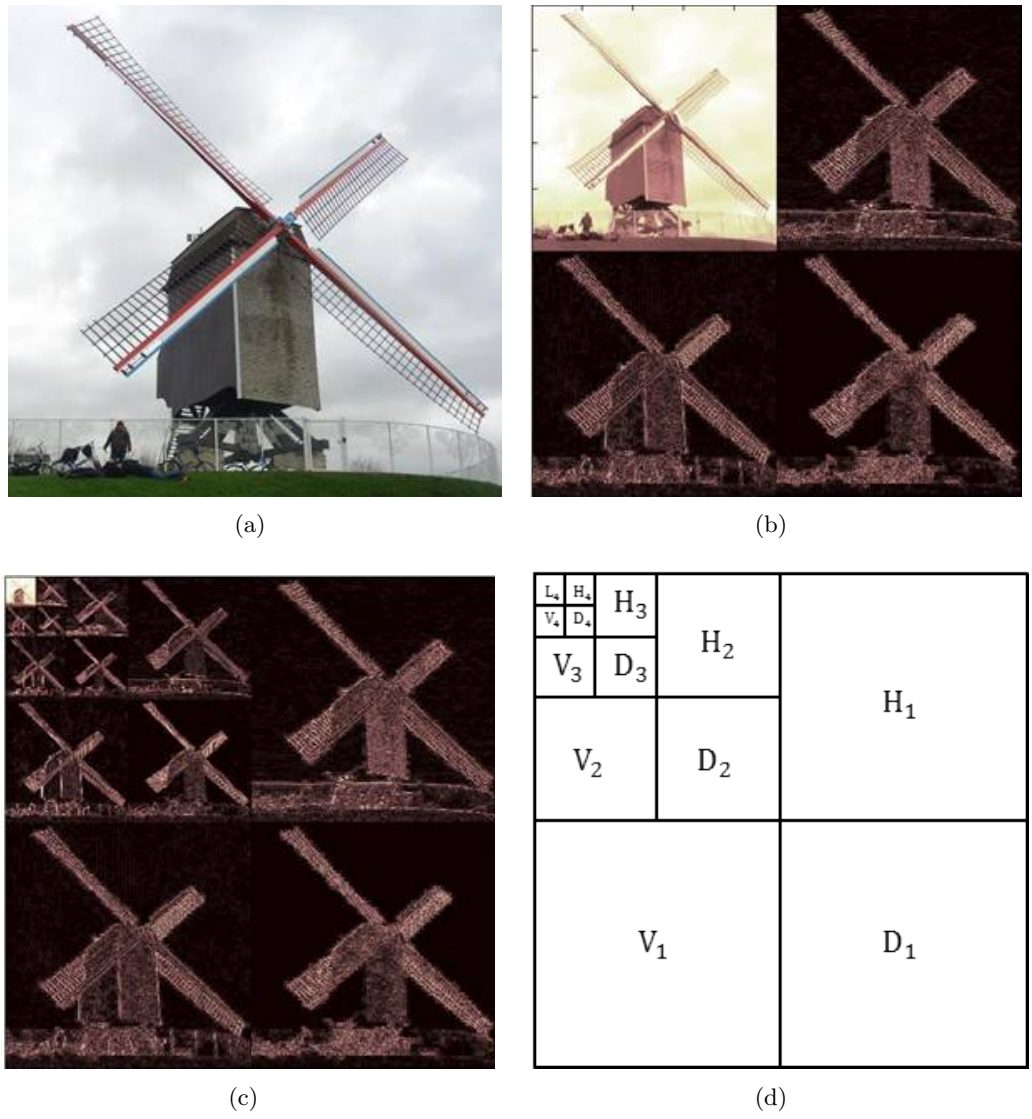


Figura A.1: Descomposición *wavelet* de una imagen

Los componentes *wavelet* de alta frecuencia de una imagen con ruido se modelan como una mezcla aditiva de una señal *Independent and Identically Distributed random variable* (IID) localmente estacionaria con media cero (La imagen libre de ruido) y el ruido blanco gausiano estacionario (El componente del ruido).

Anexo B

Libsvm

Existen muchas implementaciones de la Máquina de Vectores de Soporte. Sin embargo, en este trabajo se optó por utilizar la librería libSVM para realizar el proceso de clasificación de las imágenes ya que incluye una *Application Programming Interface* (API) para cada uno de los lenguajes de programación más populares y más utilizados en el procesamiento de imágenes digitales, tales como C, Python y Matlab [CL].

libSVM proporciona al usuario diferentes herramientas para la clasificación, e incluye utilidades que permiten la clasificación en múltiples clases. Una tarea de clasificación incluye las siguientes etapas.

- Preparar datos característicos de los ítems a clasificar.
- Separar datos de entrenamiento y predicción.
- Transformar los datos en formato de entrada adecuado para LIBSVM.
- Escalar los datos.
- Probar diferentes kernels (el más habitual RBF).
- Usar validación cruzada para encontrar los parámetros C y γ.
- Utilizar los parámetros óptimos C y γ para entrenar SVM.
- Predecir los datos.

Las características que se proponen seleccionar, corresponden a la aplicación de los diferentes filtros a la imagen, explicados en la sección 4.3, teniendo así los valores característicos para los cuales se aplicará la clasificación.

B.1 Preprocesamiento de la Información

Cada instancia de datos, está representada por un vector de números reales, tanto los valores característicos como el valor de la clase en sí.

B.2 Escalado

Es importante tener en cuenta este punto, el objetivo de escalar los datos es evitar que los atributos estén en un rango numérico grande, en vez de uno menor, ya que los cálculos son más complicados. Se utiliza el mismo método de escalado para los datos de entrenamiento y los de testeo.

B.3 Kernel RBF

El kernel realiza la separación y traslado de las muestras al espacio de características ≈ producto escalar genérico. El kernel RBF hace corresponder de un modo no lineal ejemplos en un espacio de dimensión mayor, tiene menos hiperparámetros (C y γ) que el polinomial (menor complejidad de modelo) y menos problemas numéricos.

$$0 < K_{ij} \leq 1 \tag{B.1}$$

Los polinomiales están entre 0 e infinito y el sigmoideo no es válido bajo algunos parámetros.

B.4 Validación Cruzada y Búsqueda en Rejilla

Se trata de buscar los mejores C y γ para clasificar con precisión datos de testeo. Por ello, es conveniente usar validación cruzada sobre los datos de entrenamiento, y así se evita el sobreentrenamiento.

Anexo C

Implementación de los Algoritmos

C.1 Lenguajes y Librerías

Para la implementación de los algoritmos se ha utilizado el lenguaje Python y posteriormente se pasó en su mayoría a C para conseguir las características de las imágenes en un menor tiempo ya que Python era lento, sobre todo cuando se trataba con imágenes grandes.

Para la implementación se utilizaron las siguientes librerías de Python:

- Numpy: librería Python para la realización de cálculos matemáticos. Proporciona una potente clase para manejar arrays multidimensionales compatible con C.
- Scipy: extensión a Numpy que proporciona funciones de diferentes campos científicos. Contiene funciones útiles como la varianza y la media.
- Pywavelets: librería Python usada para descomponer la imagen en sus componentes wavelet. necesarias para la función de eliminación de ruido y para la extracción de las características de la imagen.
- PythonImaging Library: librería que facilita el manejo de imágenes en python. Admite diversos formatos de imagen e incluye funciones para el procesamiento de imágenes.
- libSVM: implementación de SVM muy rápida y robusta que permite trabajar con más de dos clases (ver Anexo B).

C.2 Mejora de Rendimiento con Lenguaje C

Durante la implementación del código para la extracción de las características de las imágenes se observó que debido a la gran cantidad de datos que hay que manipular para la obtención de dichas características (para cada característica hay que procesar todos y cada uno de los píxeles, por ejemplo, en una foto de 5 mega píxeles tenemos que hacer cálculos

con 5 millones de píxeles) Python se demoraba en exceso. Esto se debe a la lentitud del lenguaje al realizar cálculos en los que hay ciclos anidados.

La solución a este problema es el uso del lenguaje C en aquellas partes del código con gran complejidad de cómputo, ya que este lenguaje al ser código compilado y no utilizar un intérprete como Python consigue un rendimiento muy superior en cuanto a velocidad de cálculo.

A todo esto y para incrementar aún más el rendimiento se paralelizaron la obtención de características, aprovechando los sistemas multinúcleo actuales.

Para combinar código C y código Python es necesario realizar cuatro pasos:

1. Crear el código Python desde el que se llamará a la función o funciones implementadas en C.

2. Mandar los datos al módulo C haciendo una llamada a las funciones de dicho módulo.

3. Dentro del módulo se debe tener el algoritmo para procesar los datos.

4. Una vez que se han procesado los datos y generado una solución, enviar el resultado de regreso a Python.

C.2.1 Implementación del Módulo

A continuación se muestra un ejemplo de un módulo implementado en C:

```
#include < Python.h >
#include < numpy/arrayobject.h >
#include < math.h >

static PyObject *funcionesC_structuralContent(PyObject *self,PyObject *args){
PyObject *imOObject,*imSObject;
PyArrayObject *imagenO,*imagenS;
PyArg_ParseTuple(args,"OO", &imOObject, &imSObject);
imagenO = (PyArrayObject *)imOObject;
imagenS = (PyArrayObject *)imSObject;
int alto = PyArray_DIMS(imagenO)[1];
int ancho = PyArray_DIMS(imagenO)[2];
int i,j;
double sumRO=0, sumGO=0, sumBO=0, sumRS=0, sumGS=0, sumBS=0;
for(i=0;i¡alto;i++){
    for(j=0;j¡ancho;j++){
        sumRO = sumRO + pow((*(unsigned char*)PyArray_GETPTR3(imagenO,0,i,j)),2);
        sumGO = sumGO + pow((*(unsigned char*)PyArray_GETPTR3(imagenO,1,i,j)),2);
        sumBO = sumBO + pow((*(unsigned char*)PyArray_GETPTR3(imagenO,2,i,j)),2);
        sumRS = sumRS + pow((*(unsigned char*)PyArray_GETPTR3(imagenS,0,i,j)),2);
        sumGS = sumGS + pow((*(unsigned char*)PyArray_GETPTR3(imagenS,1,i,j)),2);
        sumBS = sumBS + pow((*(unsigned char*)PyArray_GETPTR3(imagenS,2,i,j)),2);
    }
}
double scR = sumRO / sumRS;
double scG = sumGO / sumGS;
double scB = sumBO / sumBS;
```

```
return Py_BuildValue("ddd",scR,scG,scB);
}

static PyMethodDef funcionesCMethods[] = {
{"structuralContent", funcionesC_structuralContent, METH_VARARGS,
"Structural Content"},
{NULL, NULL, 0, NULL}
};


void initfuncionesC(void){

(void) Py_InitModule("funcionesC", funcionesCMethods);

}
```

Este código es obtiene lo que se le envía desde Python en la función *funciones_structural Content* a través del parámetro args, que es una tupla de Python con todos aquellos parámetros que se desean mandar a la función. Para separar los datos contenidos en esta tupla se utiliza la función *PyArg_ParseTuple()*, que tiene como entrada a args, una cadena de texto con los tipos de las variables y finalmente un puntero a las variables de C donde queremos almacenar los parámetros de entrada. En este caso, al tratarse de dos objetos de tipo *ndarray* de la librería *numpy* se pone como tipo la cadena "OO" ('O' es para objetos, 'i' para enteros, 'd' para doubles, etc.) y los se almacenan en variables de tipo *PyObject* para posteriormente hacer un casting a *PyArrayObject*. Una vez hecho es posible utilizar las funciones de la clase ndarray con los datos de entrada.

Posteriormente se tiene el código C que ejecuta el algoritmo con los datos.

Finalmente se manda el resultado a Python. Para ello se usa la función *Py_BuildValue()* que toma como parámetros una cadena de texto con los tipos de los datos a devolver y las variables que contienen dichos datos, análogo a como se hacía con *PyArg_ParseTuple* pero al revés, esta vez lo que devuelve es un objeto *(PyObject)* de tipo tupla. En el código de ejemplo se devuelve una tupla con tres valores de tipo double.

Dentro del módulo también se debe añadir la tabla de métodos, en la cual se incluyen todos los métodos que se desean ejecutar desde Python de la siguiente manera:

{*"Nombre de función que usaremos en Python", Nombre de la función en C, METH_VARARGS, "Comentario para explicar el uso de la función"*}

Finalmente, se tiene que crear una función con nombre *initNombreDelModulo()*, que se encarga de inicializar el módulo con la tabla de métodos que se ha creado antes.

C.2.2 Compilación del Módulo y Uso en Python

Para poder utilizar el módulo antes se debe compilar. Para ello se crea un archivo setup.py como se muestra a continuación:

from distutils.core import setup, Extension

module = Extension('funcionesC', sources = ['funcionesC.c'])
setup(name = 'Prueba C', version = '1.0', ext_modules = [module])

Dentro de este archivo se introduce el nombre del módulo ("funciones") y el nombre del archivo a compilar ("funcionesC.c"). Ahora desde la consola, con el código C en el mismo directorio se escribe:

python setup.py build

Esta instrucción creará una carpeta *"build"* en el directorio, y dentro una carpeta llamada *"lib.SO-VerPython"* donde SO es el SO con el que se cuenta y *VerPython* es la versión de Python para la que esta compilado nuestro módulo. En esta carpeta se encontrará el archivo *.pyd*. Hasta este punto sólo resta importar el módulo desde el código python.

Para importar el módulo desde python se hace un *import*. En este caso *"importfuncionesC"* y a partir de ese momento ya se pueden utilizar todas las funciones implementadas en él.

Anexo D

Sistemas de Clasificación

En este capítulo se estudian los elementos que componen un sistema de clasificación y los tipos de aprendizaje existentes, centrándonos en el aprendizaje supervisado. Posteriormente se presentan las diferentes técnicas de clasificación para la distinción de clases mediante características y se describe en detalle en qué consiste la Máquina de Vectores de Soporte (SVM) que es la más utilizada en el análisis forense de imágenes.

D.1 Introducción

La clasificación consiste en predecir el valor de un atributo (clase) basándose en los valores de otros atributos (los atributos predictores). La clasificación puede ser formalizada como la tarea de aproximar una función objetivo desconocida [DHS01].

Los principales componentes de un sistema de clasificación general son:

- **Clase**: Conjunto de entidades que comparten alguna característica que las diferencia de otras.
- **Clase de rechazo**: Conjunto de entidades que no se pueden etiquetar como ninguna de las clases del problema.
- **Extractor de características**: Subsistema que extrae información relevante para la clasificación a partir de las entidades cuantificables.
- **Clasificador**: Subsistema que utiliza un vector de características de la entidad cuantificable y lo asigna a una de las clases.
- **Evaluación de error de clasificación**: Error que se comete al realizar la clasificación.
- **Falso negativo y Falso positivo**: es el error producido cuando el sistema diagnostica una clase siendo la otra la correcta. Para problemas con dos clases, estas definiciones reflejan la importancia de una decisión contra la opuesta.

El diseño de un clasificador comprende dos pasos:

- Definir el conjunto de etiquetas
- Definir qué salidas debe proporcionar el sistema

Una vez establecido el conjunto de clases se procede a la construcción del clasificador. Esto conlleva las siguientes etapas:

1. Elección del modelo
2. Aprendizaje o entrenamiento del clasificador
3. Verificación de los resultados

Formalmente, un clasificador es un elemento que proporciona una clase etiquetada como salida a partir de un conjunto de características tomadas como entradas. Una manera de construir un clasificador es coger un conjunto de ejemplos etiquetados y tratar de definir una regla que pueda asignar una etiqueta a cualquier otro dato de entrada.

El Aprendizaje automático es la parte básica que tienen en común los diferentes tipos de clasificadores que existen. La idea básica del aprendizaje consiste en utilizar las percepciones no sólo para actuar, sino también para mejorar la habilidad de un agente para actuar en el futuro. Existen diferentes tipos de técnicas de aprendizaje.

D.1.1 Aprendizaje Supervisado

El aprendizaje supervisado consiste en aprender una función, a partir de ejemplos etiquetados anteriormente, que establezca una correspondencia entre las entradas y las salidas deseadas del sistema. No siempre es posible hacer este tipo de entrenamiento ya que tiene que disponer de la salida esperada en la función de entrada. El sistema de aprendizaje trata de etiquetar (clasificación) una serie de vectores utilizando una entre varias categorías (clases).

D.1.2 Aprendizaje no Supervisado

El aprendizaje no supervisado consiste en aprender a partir de patrones de entradas para los que no se especifiquen los valores de sus salidas. El principal problema de esta técnica es la toma de decisiones a la hora de escoger un patrón entre todos los proporcionados. El sistema trata los objetos de entrada como un conjunto de variables aleatorias, construyendo un modelo de densidad para el conjunto de datos.

D.1.3 Aprendizaje Semi-Supervisado

Actualmente existen técnicas que combinan las dos anteriores, ya que en algunos casos puede resultar muy costoso asignar etiquetas o clases a todos los datos. La finalidad es combinar datos etiquetados y no etiquetados para mejorar la construcción de modelos. Aunque no siempre es útil y existen varios métodos para llevarlo a cabo.

D.1.4 Aprendizaje por Refuerzo

El aprendizaje por refuerzo consiste en aprender observando el entorno. La idea del aprendizaje consiste en construir una función que tenga el comportamiento observado en sus datos de entrada y de salida. Los métodos de aprendizaje se pueden entender como la búsqueda de un espacio de hipótesis para encontrar la función adecuada.

D.2 Taxonomía de los Clasificadores

A continuación se describen los diferentes tipos de clasificadores existentes:

D.2.1 Clasificadores Basados en Distancias

Estos clasificadores se basan en el concepto de distancia entre los vectores de características. Los más destacados son:

- **Clasificador de distancia euclídea determinista a priori**: Es un clasificador determinístico, supervisado y a priori. Está basado en el cálculo de un prototipo (también denominado "centroide") para cada una de las K clases en las que se divide el universo de trabajo. Este prototipo puede verse como un representante "ejemplar" de cómo debería de ser un vector de características de esa clase. Así, ante un patrón desconocido se calcula la distancia euclídea del patrón que se desea clasificar a cada uno de los K prototipos. Un patrón desconocido X se clasificará como correspondiente a la clase cuyo prototipo esté a menor distancia según la distancia euclídea. Así, el clasificador euclídeo divide el espacio de características en regiones mediante "hiperplanos" equidistantes de los centroides.

- **Clasificador estadístico a priori**: Este clasificador ofrece más garantías al tratar de separar patrones para los que no se encuentra una separación lineal. Además el clasificador estadístico proporciona probabilidades de pertenencia a las clases, por lo que se debe incluir en el grupo de los clasificadores no deterministas. En situaciones en donde los vectores de alguna clase presenten una dispersión significativa respecto a la media, o en aquéllas en las que no existe posible separación lineal entre las clases, puede ofrecer mejores resultados la sustitución de la distancia euclídea por la distancia de Mahalanobis. Esta medida, que tiene en cuenta la desviación típica de los vectores de características de los patrones de la muestra, puede proporcionar regiones de separación entre clases que sigan curvas cónicas.

- **Clasificador con aprendizaje supervisado**: En este clasificador, el problema de la clasificación de un vector X en una de K clases se plantea como un problema de optimización. Más formalmente, para un conjunto de K clases $?1, ?2, ...?K$ la solución al problema de clasificar un vector X desconocido consiste en asociarlo a aquella clase cuya función discriminante $fdi(X)$ dé un resultado máximo.

D.2.2 Clasificadores Bayesianos

Estos clasificadores se fundamentan en la regla de Bayes del mínimo error. Un objeto, con unas características determinadas, pertenece a una clase si la probabilidad de pertenecer a ésta clase es mayor que la probabilidad de pertenecer a cualquier otra clase, como se muestra en la ecuación (D.1).

$$X \in \Omega_i si P(\omega)\, \rho\left(\frac{X}{\omega_i}\right) > P(\omega_j)\, \rho\left(\frac{X}{\omega_j}\right) \tag{D.1}$$

Donde,

ω es el espacio de características que está dividido en regiones, $\omega_i, i = 1, 2, ..., N$, siendo N el número de clases.

$P(\omega_i)$ es la probabilidad a priori de la clase ω_i con características X.

ω_i y $\rho\left(\frac{X}{\omega_i}\right)$ es la función de probabilidad condicional de la clase ω_i para X.

En la práctica, las funciones de probabilidad no se conocen y por lo tanto se deben estimar. Para estimarlas, primero se asume la forma de la función de probabilidad, y luego se hallan sus parámetros a partir del conjunto de entrenamiento.

D.2.3 Clasificadores de Redes Neuronales

Las redes neuronales son una técnica de aproximación paramétrica útil para construir modelos de densidad. Esta red presenta una capa de entrada con n neuronas y una capa de salida con m neuronas y al menos una capa de neuronas ocultas internas. Cada neurona (menos en la capa de entrada) recibe entrada de todas las neuronas de la capa previa y genera salida hacia todas las neuronas de la capa siguiente (salvo las de salida). No hay conexiones hacia atrás (*feedback*) ni laterales o autorrecurrentes.

El funcionamiento de la red consiste en un aprendizaje de un conjunto predefinido de pares de entradas-salidas dados como ejemplo, empleando un ciclo propagación-adaptación de dos fases. Primero se aplica un patrón de entrada como estímulo para la primera capa de las neuronas de la red, éste patrón se va propagando a través de todas las capas superiores hasta generar una salida. Se compara el resultado obtenido en las neuronas de salida con la salida que se desea obtener y se calcula un valor del error para cada neurona de salida. Estos errores se transmiten hacia atrás, partiendo de la capa de salida, hacia todas las neuronas de la capa intermedia que contribuyan directamente a la salida, recibiendo el porcentaje de error aproximado a la participación de la neurona intermedia en la salida original. Este proceso se repite capa por capa, hasta que todas las neuronas de la red hayan recibido un error que describa su aportación relativa al error total. Basándose en el valor del error recibido, se reajustan los pesos de conexión de cada neurona, de manera que en la siguiente vez que se presente el mismo patrón, la salida esté más cercana a la deseada, es decir, que el error disminuya.

Esta técnica requiere el uso de neuronas cuya función de activación sea continua y por

tanto diferenciable.

D.2.4 Algoritmos de Agrupamiento (Clustering)

En ocasiones no existe la figura del maestro que determina los patrones que pertenecen a una clase o a otra. En estos casos los algoritmos de agrupamiento o clustering permiten realizar esta tarea de manera automática. Estos algoritmos también se conocen como algoritmos de clasificación autoorganizados.

Los algoritmos de agrupación de clases se suelen utilizar cuando no existe conocimiento a priori de las clases en que se pueden distribuir los objetos, cuando las clases no son interpretables por un humano, o cuando el número de clases es muy elevado para un procesado no automático.

Algunos ejemplos de algoritmo son:

- **Algoritmo de distancias encadenadas**: Construye una cadena partiendo de un patrón al azar y encadenando cada vez el patrón que esté más cerca del extremo de dicha cadena. El algoritmo crea automáticamente nuevas clases cuando la distancia entre dos patrones consecutivos supera cierto umbral. Para fijar la sensibilidad en la determinación de las clases, este algoritmo necesita del ajuste previo de dicho umbral.

- **Algoritmo MaxMin**: Elige paulatinamente representantes de entre los patrones de muestra, determinando la distancia a la que se encuentran el resto de patrones de los mismos. Si para algún patrón esta distancia supera cierto umbral se crea una nueva clase con ese patrón como representante. El proceso se repite hasta que no se producen cambios.

- **Algoritmo de las K-medias**: Permite determinar la posición de k centroides que distribuyan de manera equitativa un conjunto de patrones. Debe notarse que, a diferencia de los algoritmos anteriores, este algoritmo tiene la particularidad de necesitar conocer a priori el número k de clases existentes.

D.3 Máquina de Vectores de Soporte SVM

Los algoritmos SVM pertenecen a la familia de los clasificadores lineales. En estos clasificadores tienen la característica de que, a priori, se conocen las clases a las que pertenecen nuestros individuos, no se trata de una agrupación por similitudes, sino que se tiene las clases bien definidas [DHS01, Sch12].

Dado un conjunto de ejemplos de entrenamiento (muestras) se pueden etiquetar las clases y entrenar una SVM para construir un modelo que prediga la clase de una nueva muestra. Intuitivamente, una SVM es un modelo que representa a los puntos de muestra en el espacio, separando las clases por un espacio lo más amplio posible. Cuando las nuevas

muestras se ponen en correspondencia con dicho modelo, en función de su proximidad pueden ser clasificadas a una u otra clase, dependiendo de la proximidad a cada una.

Más formalmente, una SVM construye un hiperplano o conjunto de hiperplanos en un espacio de dimensionalidad muy alta (o incluso infinita) que puede ser utilizado en problemas de clasificación o regresión. Una buena separación entre las clases permitirá una clasificación más correcta.

Fue diseñada en principio para tratar problemas de clasificación binarios (en dos grupos), se trata de una máquina de aprendizaje que implementa la siguiente idea: cuando no sea posible separar los datos en el espacio de entrada con un hiperplano lineal, trasladar, mediante una aplicación no lineal, los vectores de entrada a un nuevo espacio de dimensión más alta. En este nuevo espacio se construirá una superficie de decisión lineal. Las especiales propiedades que poseerá esta superficie garantizarán que la capacidad de generalización de la máquina de aprendizaje sea alta. Aunque esta idea se empleó en los primeros experimentos para datos que podían separarse sin errores, se puede extender para el caso no separable con notable éxito. La parte conceptual del problema la resolvió Vapnik para el caso de hiperplanos óptimos para clases separables. En este contexto, Vapnik definió un hiperplano óptimo como una función de decisión lineal con el margen de separación máximo entre los vectores de las dos clases. Se observó entonces que para construir el hiperplano, uno sólo debía tener en cuenta una cantidad pequeña de los datos de entrenamiento, los llamados vectores soporte, quienes determinaban ese margen [CL].

El la Figura D.1 se puede observar ($IR_n \rightarrow \{+1, -1\}$) gráficamente.

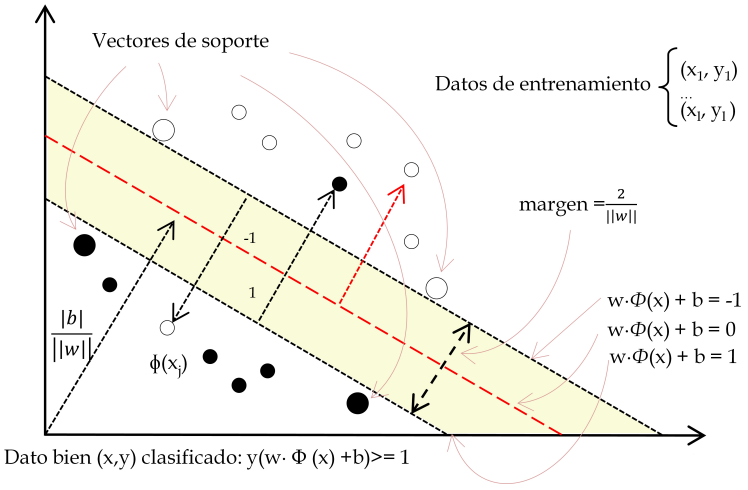


Figura D.1: Representación gráfica de máquinas de soporte vectorial

Para tener una máquina bien adaptada, debe ser entrenada con datos que pertenezcan a clases conocidas (datos de entrenamiento). Matemáticamente:

Sea X, tal que:

$$X = \{(x1, y1), ..., (xl, yl) : xi\ IR_n,\ yi\{-1, +1\}\} \tag{D.2}$$

Suponiendo que es posible separar este conjunto de entrenamiento mediante un hiperplano lineal. Los puntos x que pertenecen al hiperplano satisfacen la ecuación:

En D.2, xi son los puntos-dato e yi su etiqueta o clase (puede ser -1 o 1). La función de decisión

$$fx : IR_n \rightarrow\ \{-1,\ +1\} \tag{D.3}$$

Una forma sencilla e intuitiva para separar los datos-puntos en 2 clases de IR_2 es la construcción de una línea recta de separación y un plano de separación en IR_3. En el espacio de dimensiones superiores se habla de hiperplanos.

Suponiendo que es posible separar este conjunto de entrenamiento mediante un hiperplano lineal. Los puntos x que pertenecen al hiperplano satisfacen la ecuación:

$$w \cdot x + b = 0 \tag{D.4}$$

donde w es un vector normal al hiperplano, y:

$$\frac{|b|}{||w||} \tag{D.5}$$

es la distancia perpendicular del hiperplano al origen (se toma $|| \cdot ||$ como la norma euclídea). De entre todos los hiperplanos capaces de separar los datos, existe un único hiperplano óptimo, en el sentido de que es capaz de separar los puntos con el mayor margen de separación entre cada elemento del conjunto de entrenamiento y el hiperplano. En este sentido, el algoritmo de aprendizaje diseñado por Vapnik y Chervonenkis, las máquinas de soporte de vectores, resuelve el siguiente problema:

Encontrar $w \in IR_n$ y $b \in IR$ que minimicen:

$$\tau(w) = \frac{1}{2}||w||2 \tag{D.6}$$

Sujeto a la condición:

$$yi\ (wT \cdot xi + b) \geq 1\ \forall\ i = 1, \ldots, l \tag{D.7}$$

Cuando se hallen w y b, la regla de clasificación será, simplemente:

$$sign(wT \cdot xi + b) \tag{D.8}$$

y el error de clasificación cometido vendrá dado por R:

$$ent(w, b) \tag{D.9}$$

Por otro lado, aquellos puntos que verifican la igualdad en la inecuación

$$yi\ (wT \cdot xi + b) \geq 1 \tag{D.10}$$

La eliminación de los puntos que verifican D.7 cambiaría la solución que se encuentre, son los que se llamará vectores soporte. Estos vectores, pertenecerán a uno de los dos posibles hiperplanos óptimos de separación de los que se hablaba antes, representados para los dos hiperplanos por las ecuaciones:

$$wt \cdot xi + b = 1 \tag{D.11}$$

$$wt \cdot xi + b = -1 \tag{D.12}$$

Si se trata de aplicar el algoritmo anterior a datos no separables, no se encontrará ninguna solución factible, pues la función objetivo crece desmesuradamente. La formulación del problema, en este caso, sería:

Encontrar $IR_n \in w$ y $b \in IR$ y $\xi i,\ i = 1, \ldots, l$ que minimicen

$$\tau(w) = \frac{1}{2}||w||2 + C\sum_{i=1}^{l} \xi i \tag{D.13}$$

Sujeto a la condición:

$$wt \cdot xi + b \geq -1 - \xi i \quad y \quad \xi i > 0\ \forall i = 1, \ldots, l \tag{D.14}$$

Donde C es un parámetro que el clasificador deberá estimar.

Por último, en el caso de la SVM no lineal, se proyectan las variables de entrada en un espacio de dimensión mayor (normalmente de dimensión infinita) que aquel al que pertenecían dichas variables, y se aplica la SVM descrita anteriormente en este nuevo espacio, conocido como espacio de características. De este modo, la SVM no lineal es capaz de clasificar con una probabilidad de error dada por $R\ ent(w,\ b)$.Esa proyección se realiza utilizando las funciones núcleo (*kernel*).

La función *kernel* realiza la separación y traslado de las muestras al espacio de características. Es como un producto escalar genérico.

Las funciones *kernel* extienden la clase de funciones de decisión al caso no lineal. Se mapean los datos del espacio de entrada X a un amplio espacio de características X mediante la función ϕ y resolviendo el problema de aprendizaje lineal en X:

$$\phi : X \to X \tag{D.15}$$

La función real ϕ, no necesita ser conocida, es suficiente tener una función de *kernel* k que calcule el producto interno en el espacio de características.

$$K(x, y) = \phi(x) \cdot \phi(y) \tag{D.16}$$

El llamado "problema de selección de características", esto es, la selección de los factores o rasgos que permitan desechar aquellos elementos que se revelen como irrelevantes para el estudio que se desea realizar, ha resultado ser de especial importancia en la mayoría de los problemas de aprendizaje supervisado.

Entre los distintos modos de tratar la selección de características, el más frecuente es el siguiente: dado un conjunto de datos $(x1, y1), \ldots, (xl, yl),\ con\ x \in n,\ y\ yi \in \{-1, 1\}$ extraer un subconjunto de m variables $(m < n)$ que posean el error de clasificación menor.

Matemáticamente, la función *kernel* se define:

$$K(xi, xj) \equiv [\phi(xi) \cdot \phi(xj)] \tag{D.17}$$

Las funciones "*kernel*" más comunes son:

- Lineal $K(x_i, x_j) = (x_i \cdot x_j)$
- Tangente hiperbólica $K(x_i, x_j) = tanh(kx_i \cdot x_j + c)$ para algún $k > 0$ y $c < 0$
- Polinomial (homogeneo) $K(x_i, x_j) = (x_i \cdot x_j)^d$
- Polinomial(heterogéneo) $K(x_i, x_j) = (x_i \cdot x_j + 1)^d$
- Gaussiana RBF $K(x_i, x_j) = exp\left(-\gamma \left\|x_i - x_j\right\|^2\right)$